

SyncWare News

The journal for technical
applications of T/S
computers

Volume 5 Number 5

May.-Jun. '88



COMING EVENTS

Mark the following on your calendar, if they aren't already there:

COMPUTERFEST® '88. (This is the one that claims the right to exclusive legal use of the term COMPUTERFEST®). Hara Arena, Dayton, Ohio, August 20-21. This event covers other computers besides the Timex/Sinclair line. Sponsored by Dayton Microcomputer Association, Inc. Information from:

Mark Hanslip
143 Schloss Lane
Dayton, OH 45418
tel. (513) 263-FEST
BBS (513) 293-1754

Third Annual International/Great Northwest TS Mini-Fair, Cosmopolitan Hotel, Portland, Oregon, August 6-7. Information from:

Rod Gowen
1419-1/2 Seventh Street
Oregon City, OR
tel (503) 655-7484

TIMEX-SINCLAIR-AMSTRAD COMPUTER USERS 1988 MIDWEST REGIONAL CONFERENCE.
Beck Center for the Arts, Lakewood, Ohio, (a suburb west of Cleveland), August 26-27.
Sponsored by Cleveland Sinclair UG. Information from:

Andy Kosiorek
2192 Glenbury Ave.
Lakewood, OH 44107
Compuserve ID #75046, 3420
Cleveland Freenet BBS (216; 368-3888, ID # aa236
or c/o SYSOP Timelines T/S BBS (216) 671-6922
(10 PM-6 AM, EST)

For Your Support

SILICON MOUNTAIN COMPUTERS, C-12, Mtn. Stn. Group Box, Nelson, BC V1L 2J3, CANADA, announces:

An improved version (V 2.5) of ZX-TERM*80 is available to legitimate owners of previous versions of ZX-TERM*80. V 2.5 contains an improved high-resolution core, usable with Memotech 32K packs and unexpanded TS1500's. It also supports ASCII uploading, software-driven printer control and buffer dump, improved performance with the Byte-Back MD-2, and other refinements.

Version 3.0 of SRAM HI-RES EXTENDED BASIC is also available, to legitimate owners of previous versions. V 3.0 includes true windowing, non-destructive pull-down menus (any size), improved CIRCLE commands (including ellipses and arcs), faster window bit-scrolls, large sprites, and much more.

At this writing, both programs can be downloaded from the Nicolson Nighttime Network, (604) 354-4666. The board runs 8/N/1, and is up from 1800-0900 PDST nightly, and all day Sunday. XMODEM is supported. The programs may be available from other Timex BBSes by the time you read this; consult the file directory.

ZX-TERM*80 V2.5 is posted as ZXT25.PGM and ZXT25.VAR (program and variables files). Also available are ZXT25.DOC (instructions) and ZXT25.NAM (Hot-Z names file for hackers). The SHR-EB V3.0 core is posted as ZXSHREB.PGM. A comprehensive demo of the new features is posted as ZXSHRDEM.PGM.

Note: Please notice the change in the last three alpha-numerics of the ZIP code.

Complete TS1000/1500/ ZX81 Library for just \$10

Exploring TS1500/1000 Graphics
The Elementary Timex/Sinclair
Fifty 1K/2K Games for the ZX81 and TS1000
49 Explosive games for the ZX81
Ins & Outs of the TS1000
Making the Most of Your ZX81
The ZX81 Pocket Book
Explorer's Guide to the TS1500/1000
Basics of Timex Sinclair 1500/1000 BASIC
BASIC Basics for the TS1500/1000

The book distributor didn't know where to sell Timex books anymore, so we were able to buy up a quantity of them for a great price. Now, while they last, you can get this complete collection of ten different titles for an incredible \$10, just \$1 per book. They contain lots of wonderful program examples with explanations, programming tips, and more. Act immediately...just 60 sets are available! Order item #10BK. Include \$3 for UPS shipping, \$7 US mail or Canada

Money-back guarantee if not completely satisfied!
Zebra Systems, Inc., 78-06 Jamaica Ave.,
Woodhaven, NY 11421 (718) 296-2385

BYTE POWER 1ST CLASS MAGAZINE

BYTE POWER is a highly sophisticated computerized magazine on cassette for the TS2068 and SINCLAIR SPECTRUM 48K

No longer will you have to type in long, fastidious programs... JUST LOAD AND RUN!

BYTE POWER is the ULTIMATE magazine, over 110 programs were published up to now. Most of these are in FAST MACHINE CODE! And we bring you this QUALITY programming at a very small cost:

1 ISSUE.....	\$5.99 PP
6 ISSUES.....	\$32.00 PP
12 ISSUES.....	\$55.00 PP
FIRST CLASS PACK (6 ISSUES IN ALBUM).....	\$37.00 PP
CATALOG/DEMO TAPE.....	\$2.00 PP

Also available from BYTE POWER...

THE D.U.S. V2.0 (FOR LKDOS CART.): 5 1/4"...	\$29.95
3 1/2"...	\$31.95
BYTE POWER 1st CLASS FONTS II.....	\$19.95
THE BANNER PRINTER (Works on ANY printer!).....	\$15.95
BYTE POWER 1st CLASS PROGRAMS (ZX81).....	\$9.95

BEST OF ARCADE GAMES.....	\$19.95
BEST OF STRATEGY AND BOARD GAMES.....	\$19.95
BEST OF BUSINESS AND UTILITIES.....	\$19.95

Above programs and magazines are also available on LARKEN disks, please write for prices...

Add \$2.50 per software for Shipping. ALL PRICES ARE IN U.S. FUNDS ONLY.

For more information, write to:

BYTE POWER
1748 MEADOWVIEW AVENUE,
PICKERING, ONTARIO, CANADA L1V 3G8

LARKEN PRESENTS ...

UP TO 256K RAM for your 2068

- Expand your 2068 with up to 256K of battery backed up Ram
 - Larken Operating system lets you SAVE to memory, just like cassette or disk. (Floppy disk not required)
 - All Cassette commands supported. Very Fast and Reliable.
 - Can be used with ALL existing 2068 or Spectrum software.
 - Uses the new 32K static ram chips, 62256LP or 43256LP
 - System consists of Larken Cartridge and Rear Memory Board.
- ** PRICE - MEMORY SYSTEM with 64K Ram \$129.00
- MEMORY SYSTEM with 0 K \$ 95.00

LARKEN 2068 FLOPPY DISK SYSTEM

- The most advanced Dos available for the 2068/Spectrum. LKdos uses ALL Commands such as CAT MERGE ERASE LOAD SAVE PRINT OPEN etc. Also can support RAMDISK up to 256K and Sequential / Random Access Files (with additional software). The Larken Disk Interface can handle up to 4 floppies for up to 3.2 MegaBytes of storage. Also NMI Snapshot Save Button and KEMPSTON Joystick port on interface Also 10 Extended Basic commands for Windows and Graphics.

AERCO RAMEX or OLIGER disk users can add LKdos for more commands, Ramdisk and access to all LKdos software

** PRICE - Larken Floppy Disk System	\$119.95
- Floppy Disk IF with 0 K Memory board ..	\$169.95
- Larken Disk Editor	\$ 15.00
- Sequential/Random access files	\$ 15.00
- Xmodem to Disk Modem package	\$ 15.00
- ZX-81 Floppy Interface (15 left) ...	\$ 99.95
- LKDOS for Aerco,Ramex or Oliger Disk IF	\$ 59.95

(All prices are US, Add 6% Shipping)

LARKEN ELECTRONICS RR#2 NAVAN ONTARIO CANADA K4B-1H9
(613)-835-2680

A WORD TO AUTHORS

Some budding authors may be a little uncertain about the general criteria for publishing articles in SWN. I can't blame them, because we have never, as far as I know, compiled a hard and fast set of rules. In fact, I would argue strenuously against a rigid codification of our criteria, preferring to judge each manuscript on its own merits.

Nevertheless, something goes through my head when I look at a prospective article. Here is a rambling look at those thoughts. (And if I seem to depart from my own standards from time to time. . .well, one of my favorite quotations from Emerson says something like "foolish consistency is the hobgoblin of little minds.")

So here are some of my standards, in no particular order:

The article must be valid, as far as I can tell (and I'll admit that you can put things over on me from time to time--but I'll be chagrined every time it happens). Or we may occasionally print an honestly invalid piece, in the hope that it will provoke some thought from our readers. But I'll never try to deceive you in this respect. If I think it's not valid, I'll tell you so.

Occasionally (not often) I'll print a trivial (or apparently trivial) article because I think an author has potential, and should be encouraged. I'll do this sparingly, because it may not be fair to authors who are already at a higher stage of sophistication, and because it is asking a lot of patience from our readers.

My philosophy is that almost anyone can write a program, most people can (potentially, at least) write good programs, and a surprising number can write really good programs. The trick is in KNOWING WHAT PROGRAM TO WRITE. I will give strong preference to those who write what I think are the right programs. (And please don't ask me to define "right".)

I would hope that our readers find all of our articles informative--but I would positively delight in publishing articles that inspire our readers to think, to adapt, to experiment. As a parallel: I love good music, but I'd far rather play in a reasonably good string quartet, or brass ensemble, or the village band, or sing in a church choir that has a good conductor, than listen to the New York Philharmonic. (I would rather compose good music than any of the above, but my talents are rather limited in that direction.) I would like to think that the same attitude prevails among our readers.

I believe that a program does not have to be profound in order to be worthy of publication. Case in point, my ETAMITLU, which appeared in Vol. 5, No. 4. On the face of it, this is a very trivial program, but I would like to point out the following redeeming social virtues, to mangle the language of the courts:

1. It does something useful (after the bugs, for

which I again apologize, are eliminated), and may make it possible for authors to submit manuscripts without laying out their hard cash for word processor programs that do more than the authors may need.

2. It demonstrates that there are simple ways to meet simple needs.

3. It introduces a principle that I expect will be new to a large number of our readers--namely, the use of line renumbering to simplify the process of interpolation.

I will also give preference to articles on subjects that our readers have included in their "wish lists"--assuming, of course, that they are good articles.

And, being human, I will give a strong preference to those articles that require the least editing, correction of misspellings, etc.

FORUM

Random Access From Our Readers

M. Bowers of Warminster, PA, writes: "I would like to see a crisp explanation of memory (of TS computers). . . A comprehensive write-up, with all the details in one place, would be a great help."

Here's a challenge. Much information is available, but a comprehensive tutorial would be welcome to many readers. I love the word "crisp"--bw

From Pete Kelly of Petal, MS comes the following want list:

" . . . would be willing to buy plug-in peripherals to increase the real world usefulness of my 2068. . . (a system that uses) a digitizer and a ramdisk to create picture stories for children or information packets for adults. . . "(He suggests the city map as an example.) ". . . an article on how to hook my 2068 and 2040 printer to a bettery and small DC monitor or TV that I can carry in a briefcase or small suitcase."

Pete also calls our attention to a mini-series of articles on digitizers in RAMTOP, newlsetter of the Greater Cleveland Sinclair User Group, edited by James Dupuy, 6514 Bradley Ave., Parma, OH 44129.

Warren Fricke, from Depew, NY endorses the idea of adding voice markers to your tapes of programs, but suggests a voice announcement at the end of each program as well, both of the announcements to refer to a written log that is filed with the tape.

From Roger Phelps, of Ithaca, MI: "I would like to see a very thorough article on (how to use) the TS2068 in the extended color mode. Also I would like information on how to print and plot in the lower portion of the screen and information on sprites."

Here's another example where information is available in bits and pieces, but a comprehensive treatment of the subject would be most welcome--bw

So They Say...

The November 5, 1987 issue of the ANN ARBOR NEWS, published by University of Michigan, carries a fascinating article on artificial intelligence. One particularly provocative passage: "...there is 'extraordinary potential' now for scientists to make progress in the area of learning...INDUCTION, to be published by MIT Press later this year (1987) lays out important work that lies ahead for psychologists and computer scientists interested in thinking."

* * *

A media release from the Merrimack, NH offices of Digital Equipment Corporation (DEC) announced a "joint development effort to integrate Macintosh personal computers and AppleTalk networks with VAX systems and DECnet/OSI enterprise networks. These development efforts will take advantage of open standards for desktop integration, based on the industry-standard Open Systems Interconnect (OSI) model of the International Standards Organization (ISO)." *****

Silicon Mountain Computers to Close Doors

Silicon Mountain Computers (SMC) is closing its doors, and will no longer supply commercial software or hardware products. Existing obligations will of course be filled, and Fred will continue to be involved with the ZX81 family of fine computers; as a user, writer, and hacker, but not as a supplier.

We gratefully thank the many individuals who have supported us in the past, and made it possible to fight a losing battle for as long as we have. We appreciate the support, friendship, and patience you all have extended over the past six years.

Silicon Mountain Computers will be renamed "Silicon Mountain Electronics", which, as the name implies, will pursue avenues of a more general electronic nature as well as computers.

With the exception of SRAM HI-RES EXTENDED BASIC and ZX-TERM*80, all of the programs authored by Fred Nachbaur are hereby released from copyright restrictions. Pass them around to your friends, submit them to your user group library, enjoy them, learn from them. With the exception of the two afore-mentioned programs and any new releases that may be developed in the future, there are no strings attached. (However, any "shareware" contributions would certainly not be refused.) The two excluded programs and any new releases will be offered for distribution to established retailers.

We encourage other programmers and developers who are no longer actively marketing their products, to do the same. There are many people who can still benefit from your work, and you will be doing them a big favour by letting them do, with your blessing, what they're doing already anyway. (Besides, it's a cheap way to attain a modicum of immortality.)

As the rumour mill has already reported, one of Silicon Mountain Electronics' research projects is the investigation of the feasibility of developing a new computer. It should be clearly understood, however, that this project is by no means a certainty at this point. It's not because of the infamous "big IF", rather it depends on a whole lot of "little ifs". I have carefully chosen a core of potential developers who have expressed an interest to investigate the potentials; IF we all agree on the route to take, IF we all find the time to do our parts, IF the economics fall into place, IF the result of our brain-pooling results in a marketable product, IF no-one comes up with a better mouse before we build a better trap. . . then there will be a new computer. But don't believe anything you hear, unless you hear it from us.

If it does happen, it will not be, as rumour has it, a Timex "clone". The new machine will have some common features, such as elegance in simplicity, but it will be a new machine in its own right. We'll keep you posted.

EDITOR'S COMMENT: We are sorry to see Fred leave the computer field--or rather, to let his interest in computers be diluted. I don't know how many of you realize how much the Timex fraternity owes to this one man.

His release of programs to the public domain is typical of Fred's style--he wants the world to reap whatever residual benefits it can from his brain-children, even though he will no longer be nurturing them.

As any author/programmer/composer or other artist knows, Fred's remark about immortality, though light-hearted, is quite serious. Let's help him get the immortality he deserves, by continuing to acknowledge his authorship on any of the programs we appropriate with his permission.

Good luck, Fred. We trust that we haven't heard the last from you. We are sure that you haven't heard the last from us.

Cedric Bastiaans, SWN Author, Dies

I was informed that Cedric Bastiaans died on April 30.

I never met Cedric personally, but had got to know him pretty well through correspondence over the past few months. Maybe some of you remember him as a contributor to SyncWare News some years ago. I believe that he has also been active in a number of users' groups.

Cedric told me early on that his heart was failing, and he was forced accordingly to cut down on his output. He accepted the limitation realistically, however, without self-pity and without losing his good humor. He was an accomplished linguist and (as is not at all uncommon with linguists) his programming showed a deep insight into the workings of the human mind. He will be sadly missed.

Condolences may be sent to

Ernest D. Bastiaans
3 Cassie Ct.
Mt. Sinai, NY 11766

Lemke Software Development

QUALITY PRODUCTS FOR THE TS-2068

Pixel Print PLUS! THE DESKTOP PUBLISHER by Lemke Software

What's the PLUS?
PERFORMANCE!

Checkout these SPECS:

- 1) WYSIWYG (What You See Is What You Get!)
Create your text on the screen... as easy as typing!
LOAD graphics... Load ICONs... Import TASWORD text via the TASWORD conversion utility.
- 2) AUTOMATIC and MANUAL line and character adjustments.
- 3) RESTORE FONT (after using the BOLD/MODERN/ITALIC modifiers).
- 4) KEEP/UNDO/SAVE/LOAD/LOAD ICON LOAD SCREENS/SAVE SCREENS WIDE/HIGH/CLS/SCROLL SPEED
- 5) OVER/INVERSE/CAPS LOCK UP TO 16 POINT FONTS (font package in development)
- 6) COPY/ERASE/INSERT/DELETE/NEW
- 7) AERCO/TASMAN A&B/A&J CPI
- 8) IBM/EPSON/PROWRITER type printers.

Pixel Print PLUS! has all of the features found in v2.0, it is 100% compatible with the Pixel Print ICON #1 & #2 packages, FONTS #1, #2, & #3 as well as the TASWORD Util. and your Pixel Print files!!



Lemke Software
2144 White Oak
Wichita, KS
67207

PIXEL PRINT PLUS!....	\$19.95
TASWORD TEXT CONV....	\$19.95
ICON PACKAGE #1.....	\$19.95
ICON LIBRARY #2.....	\$14.95
FONT PACKAGE #1.....	\$19.95
FONT LIBRARY #2.....	\$14.95
FONT LIBRARY #3.....	\$19.95
PIXEL SKETCH v2.....	\$19.95
CHECKBOOK RUBBER.....	\$19.95

Pixel Print PLUS!...
The PLUS! is 13 more functions to make the PIXEL PRINT DTP an even more powerful, easy to use program!!

the Pixel Print Plus Deal:

If you looked all over the...



you'd have a hard time beating a deal like this!

So get out your and ADD this up!

- Pixel Print PLUS!
- + TASWORD Utility
- + FONT Library #3
- + Pixel Sketch v2.0

a \$79.80 value for \$59.95 ppd.
(THIS OFFER NOT GOOD WITH ANY OTHER OFFER, CASSETTE VERSION ONLY..... OFFER ENDS JUNE 30)

Pixel Print Press...

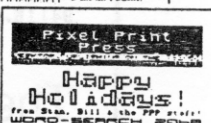
The Pixel Print Press Newsletter is a Newsletter written to give Pixel Print Desktop Publishers free USER support with a variety of tips, helps, examples, along with simple program mods to give the program extra flexibility... (like the mailing label mod).

The Pixel Print Press Newsletter is about to complete it's first year of publication! Bill and I would like to thank everyone who has written in with tips, helps, questions, or ideas. ALSO, thanks to everyone who has supported us by buying PIXEL PRINT Products!

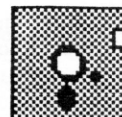
Now we are ready to begin year 2 with a NEW LOOK, more ideas, and more fun! So, don't miss out.

THE PIXEL PRINT PRESS is still a FREE publication! We publish it 4 times a year... all you need to do to get your copy is send 4 Self-Addressed-Stamped-Envelopes (business size) to Lemke Software and that's all!

Get the Pixel Print Press, today!



Pixel Print Professional AERCO DISK VERSION!

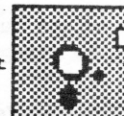


- * Pixel Print Plus v3.2
- * Create up to 20 page documents
- * Print a complete Doc header, left, & right (no more printer adjustments!)
- * Uses bank-switching to print left and right columns together
- * Print multiple Copies!
- * Newsletter Format or
- * 64 Column Letter Format

NOTE: printers must be capable of 72 to 90 Dots per inch (576 to 720 dots per line) Bit Image Graphics!!!

- * Automated Printer Customizing
- * 5500 diskette (OR AUTO-SAVE TAPE)
- * only \$29.95 ppd. (ORDER NOW!)

OLIGER DISK VERSION!



- * Same as the AERCO version above except for use with the Oliger Disk Interface

- * Requires a minimum of 32K RAM expansion... (see the 32K RAM Cartridge below!)
- * Program comes on CASSETTE and AUTO-SAVES to DISK, ready to go!!!
- * only \$29.95 ppd. (ORDER NOW!)

Pixel Print Professional

For CASSETTE & MICRODRIVE



- * Same as Aerco except for use with Tapes...

NOTE: Specify either TAPE or MICRODRIVE...

- * Requires a minimum of 32K RAM expansion... (see the 32K RAM Cartridge below!)
- * only \$29.95 ppd. (ORDER NOW!)

32K RAM CARTRIDGE



- * 32K Volatile Memory
- * Fits into the TS-2068 Cartridge Dock
- only \$40.00 plus \$4 shipping and handling

Ram cartridge price will vary with cost of memory... \$40 is firm thru June 30, 1988 (WRITE)

- * RAM Cartridge is accessed via BANK-SWITCHING... and is NOT limited to use with the PIXEL PRINT PROFESSIONAL!

Watch for other Lemke Software to use this RAM expansion!!!

Lemke Software Development

QUALITY PRODUCTS FOR THE TS-2068

Late News on Cleveland 1988 Mid West T/S Conference

The Cleveland Mid-West T/S Conference will be held on Saturday and Sunday, August 26 and 27, 1988, at

Beck Center for the Arts
Lakewood, Ohio

There is no nearby hotel complex, the organizers warn us, so you will need either a car or access to a car pool.

Inquiries should be addressed to

Andy Kosiorek
Cleveland Sinclair Users Group
2192 Glenbury Ave.
Lakewood, OH 44107

COMPUSERVE ID #75046,3420
Cleveland Freenet BBS (216-368-388, ID #aa236
or
c/o SYSOP, Timelines T/S BBS (216-671-6922 10 PM
to 6 AM, EST)

In the fully spelled out version of their title, the organizers include Amstrad along with Timex and Sinclair.

The conference welcomes exhibitors (\$25 per table), people willing to make a conference presentation, and members of the Great American Public who want to observe and learn. And to buy.

A Plug for SNUG

I hope that the proposal for a North American Users' Group (S.N.U.G.) is still alive by the time this issue of SyncWare News reached you. I also hope that it's not too late to make one more suggestion as to how SNUG can serve its members.

One of the problems that bedevil the organizers of computer shows is the danger of conflicting with other events. Shows scheduled too close together (either in space or in time) can cause difficulties both for exhibitors and for the public. A national group could help to minimize such conflicts.

I hope that we don't ever see the day when any organizer has to get permission from the national group in order to stage a show, or when any exhibitor has to get permission to take part. (And Heaven help us indeed if we ever have to get permission from anybody--except our spouses--to attend a show!) But SNUG could certainly act as a clearing house for information, so that anybody causing a conflict would at least know what he was doing.

Missing Person Report

Can anybody tell us where to find Mike Trivisano? Mike contributed a couple of fine articles to us a few years ago, on the subject Building a 2068 Data Base. I don't know why the series was discontinued, but I hope it wasn't as the result of something that we did. (Or didn't do.)

Anyway, one of our readers would like to see more of the series, and we'd like to ask Mike whether he wants to continue it. Unfortunately, the U. S. Mail tells us that they can't find him.

Can any of you let him know that we'd like to hear from him?

Truth or Consequences

You may remember a reference in our Vol. 5 No. 3 to the city of T or C, New Mexico. T.G. Morley sends us a Chamber of Commerce announcement explaining that the city, originally known as "Hot Springs", was named Truth or Consequences in 1954--to mark the 10th anniversary of the radio program called "Truth or Consequences". Three subsequent referenda have upheld the name change. In recognition of this gesture, Ralph Edwards, producer of the show, has returned to the city annually.

The Chamber also notes that the city is no longer confused with other locations with "Hot Springs" in their names, that the mineral baths and fresh air are still available, and that living costs are "extremely reasonable".

Further Info on that "Other" PC

Fred informs us that the IQ8300 is exactly the same as the PC8300. So is the Power 3000. Makes one wonder why the proliferation of names.

A Reader Replies

Earl Dunnington, of Boynton Beach, FL, volunteers the following information:

... For Gerard Triptree of Little Ferry, NJ:

An article on modifying the TS 1500 for connection to a monitor was published in "THE BEST OF TIME DESIGNS VOLUME ONE".

(NOTE: Our own SWN also treated this subject in Vol. II No. 5, in the fourth paragraph of "TS1500 VIDEO TOPICS" on page 22. -- bw.)

A TS 1500 schematic can be obtained from Sunset Electronics, 2254 Taraval Street, San Francisco, CA 94116.

The TS2068 Bookshelf

... For Dorothea Bundy of Ventura, CA:

A BASIC programmer's library for the 2068 should contain the following books:

THE 2068 Personal Color Computer USER MANUAL
THE TIMEX SINCLAIR 2068 EXPLORED--Tim Hartnell
TIMEX SINCLAIR 2068 BEGINNER/INTERMEDIATE
GUIDE--Fred Blechman
T/S 2068 BASICS AND BEYOND--Sharon Zardetto Aker
THE BASIC HANDBOOK--Encyclopedia of the BASIC
Computer Language--David A. Lien (2nd
edition and 3rd edition)

There is a real dearth of any books for machine code on the 2068. I would advise anyone learning Z80 Assembly Language to obtain a TS 1500 (or 1000) to use until you become proficient.

Reference books should include:

The appendix of the USER MANUALS

MASTERING MACHINE CODE ON YOUR TIMEX SINCLAIR
1500/1000--Toni Baker
A PRACTICAL GUIDE TO MACHINE LANGUAGE
PROGRAMMING ON THE TIMEX/SINCLAIR 1500 and
1000--David B. Wood
THE COMPLETE TIMEX/SINCLAIR ZX 81
ROM DISASSEMBLY--Dr. Ian Logan and Dr. Frank
O'Hara
INTRODUCTION TO 2068 MACHINE CODE--Dr. Lloyd
Dreger
THE TIMEX/SINCLAIR 2068 ROM MANUSCRIPT--Dr. Lloyd
Dreger
Z-80 and 8080 ASSEMBLY LANGUAGE PROGRAMMING--
Kathe Spracklen
HOW TO PROGRAM THE Z80--Rodnay Zaks (Radio Shack)

Support Available for the Z88

Larry Chavarie of Ottawa reports that there is a
Z88 Users' Club (they even put the apostrophe in
the right place!), with a bi-monthly newsletter.
The club is:

Z88 USERS' CLUB
68 WELLINGTON STREET
LONG EATON
NOTTINGHAM NG 10 4NG
ENGLAND

Their newsletter is called Z88 EPROM.

OFF THE WALL

Fred must have been caught in a Chinook when he
thought up these Daffynitions:

PEEK (2) - what happens to your interest when costs
GOSUB-normal. (It FLAGS when they GO TO NEW
PEEKs).

CODE - a bacteriud infecshud causigg a stuffy
dose.

ZX - what Maurice Chevalier would call a former
spouse.

NEXT X - what Mickey Rooney says when signing
alimony checks. (Is this also the origin of SIGNUM?
- ed.)

RND - Research and Development

CHIP - what it's ok to be off the old BLOCK, but
not ok to have on your shoulder.

BAUD - slang for curvaceous females (as, "What a
baud!")

(ed. note: is this what Shakespeare had in mind
when he used the term to mean a woman of easy
virtue?)

BAUD RATE - number of sunbathers you pass per
minute while cruising the beach.

PARITY - a satirical play or novel. Also, a
daffynition. (Ed. note: this year, the two major
parities come with elephants and donkeys.)

STOP BIT - placed in a horse's mouth to make him
STOP RUNning. (Does not work with elephants and
donkeys and their representatives--there is nothing
that will STOP them until November--ed.)

I/O (1) - what the Lone Ranger tells Silver

I/O (2) - What the 7 dwarfs (and commuters with a
mortgage) say on the way to work.

BINARY - a factory that wottles bine.

PORT (1) - one of the BINARY's products.

CARRIAGE RETURN - taking back the rented buggy
after the PROM.

PAUSE 4E4 - LINE's club slogan.

FOR (2) - pertaining to the LINE's front PAUSE.

TO - number of FOR PAUSE per LINE.

FOR (3) - golfer's shout after a HARD DRIVE.

DISK DRIVE - golfer hitting a circular divot
instead of the ball.

PC BOARD - tired of IBM's. (In Boston, this is
pronounced PC BAUD-ed.)

MOS - grows on the north side of trees.

CMOS - grows on the north side of submerged trees.

LIST - to lean, as a boat after getting snagged in
CMOS.

PORT (2) - one of the TO possible directions to
LIST.

PORT (3) - where LISTing ships head for.

RESISTOR - what every good sailor does on meeting a
siren with a good BAUD.

MEMORY BLOCK - er . . . I can't remember.

BLOCK MARKER - a street SINE.

LET - rent, as in "TO LET".

POKE (2) - (usually with "along") to move SLOWly.

POKE (3) - what you do to make a SLOW POKE move
FASTER.

ON ERR - where you're walking when your head is in
the clouds.

LARKEN 3 1/2" DISK SYSTEM for \$199.95

Package includes ...

- Larken 2068/Spectrum Disk Interface and LKdos Cartridge
- 1 - 400K 3.5" Mitsubishi Disk Drive and Power supply
- or 1 - Single sided 5 1/4" Disk drive and Power supply
- 1 Disk of Disk converted programs to start you off.
- Other Drive combinations available .
- Ready to Run Floppy disk System . 90 day guarantee
- Limited Time Offer, Until 3.5" 's are gone .

(prices are US \$ add \$10 S&H)

LARKEN ELECTRONICS, RR#2 NAVAN ONTARIO, K4B-1H9 CANADA
(613)-835-2680

INSTANT SORTING

Robert C. Fiscner

TS1000 and TS2068
Machine code

In the May-June '86 issue of SyncWare News, I went through several sort routines for the 2068 and 1000 computers. For general use, I decided that the Shell-Faulk was the best all-round choice, but I also asked readers to send in any better ideas. One reader did, Larry Crawford of London Ontario, Canada. He took the Shell-Faulk and converted it to machine code. Between the two of us, we have worked out bugs, shortened it, and made it more flexible. This routine will work on either the 2068 or Spectrum, but it should only take a little hacking to convert to a 1000.

conSORTING with Speed

Let's get right to the issue of speed. Below is a comparison of the Shell-Faulk in BASIC and machine code at sorting files which are each 10 characters long.

	BASIC	MACHINE CODE
50 files =	11.8 sec	.2 sec
100 =	29.9	.3
200 =	74.8	.7
400 =	184.0	1.8
800 =	430.0	4.9
1600 =	1049.0	15.0
3200 =	2402.0	47.0

Maybe instant isn't the best word for it, but I think most people would be happy with this much speed. It works with any two dimensional string array (this is not for numbers - sorry) with any name and you can change the dimensions at will. A few BASIC statements relay certain key bits of information to the machine code before it is called.

Sorting out the Code

First type in the following lines of BASIC:

```
10 LET X=100: LET L=10: DIM N$(X+1,L): FOR Z=1 TO X: FOR N=1 TO L: LET N$(Z,N)=CHR$(INT(65+RND*26)): NEXT N: NEXT Z: PRINT "READY": BEEP .5,40: PAUSE 0
20 POKE 23296,X-256*INT(X/256): POKE 23297,INT(X/256): POKE 23298,L-256*INT(L/256): POKE 23299,INT(L/256): LET N$(X+1)=N$(X+1): POKE 23300,PEEK 23629: POKE 23301,PEEK 23630
30 LET N$(1)=N$(1): RANDOMIZE USR 61402
100 BEEP .5,30: FOR Z=1 TO X: PRINT N$(Z): NEXT Z: STOP
9990 CLEAR 61389: LOAD "sort"CODE 61390: GO TO 1
9999 SAVE "SORT" LINE 9990: BEEP .5,40: SAVE "sort"CODE 61390,24
9: BEEP .5,40: CLS: PRINT "REWIND TO VERIFY": VERIFY "SORT": VERIFY "sort"CODE 61390,249
```

Line 10 just sets up some random files for testing purposes as was done in the BASIC version of the sort. One change is that we dimension N\$ for one extra file space. Thus, if X=100 we actually dimension N\$(101,L). This extra space is only used as a temporary storage area as files are moved. Never use it to store data. Be sure to set up this extra file space in your own programs too. As you test this program, feel free to set X (the number of files) and L (the length of files) to any values the computer has room for. If your own programs do not use N\$ for storage, just change all references to N\$ to be the name you want. The same applies to the use of X and L.

Line 20 begins the part of the routine you use in your own programs. It sets up the machine code pointers beginning with POKE 23296 and 23297 with the number of files to sort (X) not including the extra file. You could POKE a lower value than X if you only wanted to sort some of the files such as the first 50. Next we POKE the length of each file into 23298 and 23299. The two storage areas for size and length are to allow values larger than 255. This way you can have thousands of values of considerable length.

The statement LET N\$(X+1)=N\$(X+1) is used to make the computer store the address of the extra file in a special place called "dest". This is found at 23629 and 23630. Therefore we take the contents of those addresses and put them in 23300 and 23301 so the machine code can access it later.

All these POKES are made into the printer buffer as are all variables created by the machine code itself. This is one reason the program is completely relocatable. If you do relocate the machine code, do not change the location of any POKES!

file to sort, but instead of poking the location as before, we let the machine code access "dest" directly. You can make the routine start sorting after the first file if you wish. To start with the 11th file, just make the statement read: LET N\$(11)=N\$(11). Also be sure you have adjusted the total number of files to sort as well. If X=100, then by starting at file number 11, we would only sort 90 files. Without this correction, the routine would overwrite other variables. The machine code is then called and the sorting is done.

Lines 20 and 30 are the only BASIC lines you need to use this in your own programs. The actual line numbers are not important and the two lines can be combined if you wish. The remaining lines are only used to allow you to practice on this example, but are not needed in your programs. Line 100 prints out the sorted files. Line 9999 is for making a copy of the program (enter RUN 9999) and verifying it after the machine code is in place. When reloaded, line 9990 sets RAMTOP and loads the machine code. The test procedure then begins. Depending on the values you set for X and L, the setup can take quite some time - certainly longer than the sorting process. After you have saved a copy, you can test without reloading by entering RUN.

The machine code can be moved to any position you prefer as long as you have room for 249 bytes. Remember that the RANDOMIZE USR address must be 12 bytes past the start of the machine code. If you load it in at 65000, then RANDOMIZE USR 65012. Also adjust the save and load commands to reflect the change.

To enter the machine code, first enter CLEAR 61389.
Now enter the following lines which will load the machine code beginning at 61390:

```
1000 FOR N=61390 TO 61638
1010 INPUT "ENTER NUMBER ";P
1015 POKE N,P
1020 PRINT N;" ";P
1030 NEXT N
1040 STOP
```

Now enter RUN 1000 and then enter one number at a time as listed below. The numbers are listed in order horizontally so finish the first row ACROSS before going to the second line.

17	0	91	33	22	91	1	21	0	237
176	201	42	0	91	34	10	91	34	12
91	42	12	91	175	203	28	203	29	124
181	40	223	203	197	34	12	91	229	235
42	10	91	175	237	82	34	14	91	33
1	0	34	16	91	34	18	91	42	77
92	34	6	91	237	91	2	91	193	25
11	120	177	32	250	34	8	91	237	91
6	91	237	75	2	91	26	190	56	11
32	62	11	120	177	40	4	19	35	24
241	58	20	91	61	32	11	225	34	8
91	225	34	6	91	50	20	91	42	16
91	35	34	16	91	34	18	91	235	42
14	91	175	237	82	56	140	42	6	91
237	91	2	91	25	34	6	91	42	8
91	25	24	177	42	6	91	237	91	4
91	237	75	2	91	213	197	229	237	176
42	8	91	209	193	229	197	237	176	193
209	225	237	176	42	18	91	237	91	12
91	175	237	82	34	18	91	56	158	40
156	58	20	91	254	0	32	12	42	6
91	229	42	8	91	229	60	50	20	91
42	6	91	34	8	91	229	237	75	2
91	237	91	12	91	175	237	66	27	122
179	32	248	34	6	91	225	24	159	

When all the numbers have been entered, delete all lines from 1000 to 1040 and make your backup copy before testing the program.

MC Explained

The logic of the machine code closely follows that of the BASIC version. As I explain the machine code, I will refer to the original BASIC version often so here is a listing of the sort portion of that routine so you can better follow along:

```
1050 LET M=X: LET S=M
1100 LET S=INT (S/2): IF S<1 THE
N GO TO 9900
1115 IF S/2=INT (S/2) THEN LET S
=S+1
1120 FOR N=1 TO M-S: LET J=N
1150 IF N$(J)>N$(J+S) THEN LET D
$=N$(J): LET N$(J)=N$(J+S): LET
N$(J+S)=D$: LET J=J-S: IF J>0 TH
EN GO TO 1150
1160 NEXT N: GO TO 1100
```

Here is a description of the machine code:

EFCE LD DE,5B00 The start of the machine code is at 61390. This is actually the end of the sort where the printer buffer is cleared of data. DE is set to the start of the buffer which is also the start of the machine code variables.

EFD1 LD HL,5B16 HL is set to a point in the buffer that is not affected by the mc and still contains zeros.

EFD4 LD BC,0015 BC holds the number of bytes affected by the machine code routine.

EFD7 LDIR This command moves whatever is in the address represented by HL into the DE address. HL and DE are each incremented to the next address while BC is reduced by 1. This repeats until BC=0 which also means the buffer is clear so your next LPRINT won't print any garbage.

EFD9 RET Return to BASIC (sort done)

EFDA LD HL,(5B00) The actual sort starts here from BASIC (address 61402). HL is loaded with the value of X which was poked into 5B00 from BASIC.

EFDD LD,(5B0A),HL Put X in storage for M. Same as M=X in line 1050.

EFE0 LD (5B0C),HL Also put in storage for S. Same as S=M in line 1050.

EFE3 LD HL,(5B0C) This starts line 1100. Get value of S

EFE6 XOR A Clears carry flag before RR instruction and zero the A register

EFE7 RR H

EFE9 RR L These 2 instructions get the integer value of HL/2. Thus it is the same as INT (S/2) in 1100.

EFEB LD A,H

EFEC OR L

EFED JR Z,EFCE If HL=0 (S=0) then sort is done so jump to clear printer buffer before going to BASIC

EFEF SET 0,L Set bit zero of L to make sure it is an odd value (line 1115).

EFF1 LD (5B0C),HL Store new value of S.

EFF4 PUSH HL Push S on stack.

EFF5 EX DE,HL Put S value in DE

EFF6 LD HL,(5B0A) Get M value

EFF9 XOR A Clear carry flag before subtraction

EFFA SBC HL,DE Calculate M-S

EFFC LD (5B0E),HL Store M-S

FFFF LD HL,0001 Prepare for FOR-NEXT loop

F002 LD (5B10),HL Store in N. Thus loop starts as N=1

F005 LD (5B12),HL Put in J (J=N). See line 1120

F008 LD HL,(5C4D) 5C4D is the address of "dest", so this shows where the first file is.

F00B LD (5B06),HL Store address of first file in N\$(J)

F00E LD DE,(5B02) Get the length of each file

F012 POP BC Get S value

F013 ADD HL,DE

F014 DEC BC

F015 LD A,B

F016 OR C

F017 JR NZ,F013 These lines find the address of file N\$(J+S)

F019 LD (5B08),HL Store N\$(J+S) address. See 1150

F01C LD DE,(5B06) Get N\$(J) address

F020 LD BC,(5B02) Get file length

F024 LD A,(DE) Get a character from N\$(J) file

F025 CP (HL) Compare to character in N\$(J+S) file

F026 JR C,F033 Jump if the order is okay to the preparation for NEXT N

F028 JR NZ,F068 Jump to swap routine if order is wrong

F02A DEC BC If the characters are the same then we continue to compare the two files by first decrementing the number of file characters to check.

F02B LD A,B

F02C OR C

F02D JR Z,F033 These lines check to see if BC=0, thus all characters are checked. If so then jump to NEXT N.


```

F02F INC DE
F030 INC HL
F031 JR F024    To continue comparing the two
                files, we move HL and DE to the next
                characters and jump back to compare again.
F033 LD A,(5B14) This starts the preparation for
                NEXT N.
F036 DEC A      Since the FLAG was either zero or
                one, this will make it zero or 255.
F037 JR NZ,F044 If it is 255, the flag was not
                set so jump to NEXT N.
F039 POP HL     If the flag was set, get old N$(J+S)
                address. This is the address before we did
                LET J=J+S in line 1150.
F03A LD (5B08),HL Store LET J=J+S in line 1150.
F03A LD (5B08),HL Store in N$(J+S)
F03D POP HL     Get old N$(J) - what it was
                before LET J=J-S.
F03E LD (5B06),HL Store in N$(J)
F041 LD (5B14),A The A register is zero so this
                clears the FLAG.
F044 LD HL,(5B10) NEXT N begins here. First get
                old N value.
                To next value of FOR-NEXT loop
F047 INC HL     Store new N value
F048 LD (5B10),HL Store in J too (J=N)
F04B LD (5B12),HL Put J value in DE
F04E EX DE,HL   Get M-S value
F04F LD HL,(5B0E) Clear carry flag
F052 XOR A      Subtract (M-S)-J to see if
F053 SBC HL,DE  NEXT is done.
FOR-            If J is more than M-S, then
                the
                loop is done so jump back to mc version of
                line 1100.
F057 LD HL,(5B06) If loop is not done, get N$(J)
                address.
F05A LD DE,(5B02) Get length of file
F05E ADD HL,DE   Puts HL at next file
F05F LD (5B06),HL Store new N$(J) address
F062 LD HL,(5B08) Get N$(J+S) address
F065 ADD HL,DE   HL at next file (new N$(J+S)
                address).
F066 JR F019     Go back and sort again
F068 LD HL,(5B06) Start the file swap here. Get
                N$(J) address.
F06B LD DE,(5B04) Load DE with the extra N$ file
                address which is used as D$ in the BASIC
                version.
F06F LD BC,(5B02) Get file length so we know how
                many bytes to move.
F073 PUSH DE     Save D$ address
F074 PUSH BC     Save file length
F075 PUSH HL     Save N$(J) address
F076 LDIR        Put file N$(J) in D$
F078 LD HL,(5B08) Get N$(J+S) address
F07B POP DE      Get N$(J) address
F07C POP BC      Get length of file
F07D PUSH HL     Save N$(J+S)
F07E PUSH BC     Save length
F07F LDIR        Put N$(J+S) in N$(J)
F081 POP BC      Get length
F082 POP DE      Get N$(J+S) address
F083 POP HL      Get D$ address
F084 LDIR        Put D$ in N$(J+S)
F086 LD HL,(5B12) Get J value
F089 LD DE,(5B0C) Get S value
F08D XOR A       Clear carry flag
F08E SBC HL,DE   Calculate J-S
F090 LD (5B12),HL Store new J value
F093 JR C,F033   If J is less or equal to zero,
                then go to prep fN.
F095 JR Z,F033   This starts our preparation to
                go line 1150. First get FLAG.
                Compare FLAG to zero
F097 LD A,(5B14) Jump if FLAG already set

```

```

F09E LD HL,(5B06) If FLAG is not set, we must
                save the present address of N$(J), so first
                get N$(J) address.
F0A1 PUSH HL     Save N$(J) address
F0A2 LD HL,(5B08) Get N$(J+S) address to save
                too
F0A5 PUSH HL     Save it
F0A6 INC A       Make A register hold 1
F0A7 LD (5B14),A Use it to set FLAG
F0AA LD HL,(5B06) Get N$(J) address
F0AD LD (5B08),HL Put it in N$(J+S). This
                allows for LET J=J-S.
F0B0 PUSH HL     Save address of N$(J+S)
F0B1 LD BC,(5B02) Get file length
F0B5 LD DE,(5B0C) Get S value
F0B9 XOR A       Clear carry flag
F0BA SBC HL,BC
F0BC DEC DE
F0BD LD A,D
F0BE OR E
F0BF JR NZ,F0B9  These adjust the address of
                N$(J) because of the LET J=J-S
                command.
F0C1 LD (5B06),HL Store new N$(J) address
F0C4 POP HL      Get N$(J+S) address
F0C5 JR F066     Now go back to sort again. The
                jump is too far for a relative jump so it is
                done in two steps. First to F066 which then
                goes to F019.

```

The variables are all found in the printer buffer as follows:

5B00-5B01 (23296-7)	X value from BASIC
5B02-5B03 (23298-9)	Length of file from BASIC
5B04-5B05 (23300-1)	Extra file address from BASIC
5B06-5B07 (23302-3)	N\$(J) address
5B08-5B09 (23304-5)	N\$(J+S) address
5B0A-5B0B (23306-7)	M value
5B0C-5B0D (23308-9)	S value
5B0E-5B0F (23310-1)	M-S value
5B10-5B11 (23312-3)	N value
5B12-5B13 (23314-5)	J value
5B14 (23316)	FLAG

As always, let me know if you have any better ideas (BASIC or machine code). The address is: Robert C. Fischer; 804 Old York Hwy; Apt 3-B; Dunlap, TN 37327

TYD ☆ BYTS

Fred Nachbaur takes issue with my comment about static electric shocks. He says that what you feel really is the electricity, rather than heat, as I suggested. I'm not going to argue, for two reasons:

1. Fred may be right, and
2. I may be wrong.

There's complementarity for you.

Anyway, the technique does work. Fred suggests that it's because the current density is reduced.

BUG ALERT

Somewhere in between the Editor's in-box and the printer, a piece of Shawn Byrne's World Geography program (SWN Vol. 5 No. 4) got lost. The lines of the BASIC part of the program between Line 130 and Line 1040 are still floating around somewhere-- either in that world that Shawn was quizzing us on, or in some parallel (or skewed) universe.

The missing lines, with their immediate context, are printed here, with our apologies to Shawn and to our readers. By the way, the apparent gap between Line 2100 and Line 5000 is not an accident. Shawn didn't program any lines in that area.

I probably should have pointed out that this program built on Shawn's fine GET routine, which was published in our Vol. 3 No. 3.

```

100 PRINT "DO YOU WANT TO (TAB
2;"[REVIEW]";TAB 2;"[IDENTIFY]";
AT 4,2;"PRESS 1 OR 2";AT 4,0;
110 LET G$=" "
112 IF NOT USA GET THEN GOTO 11
2
113 REM 1,2
120 PRINT AT 4,2;" "
130 LET G$=G$
190 PRINT AT 9,0;"CHOOSE";TAB 2
;"[COUNTRIES AND ISLANDS]";TAB 2;
;"[CONTINENTS, OCEANS, AND SEAS]";
AT 13,2;"PRESS A OR B";AT 13,0
200 IF NOT USA GET THEN GOTO 20
0
202 REM A,B
210 CLS
220 LET C$=G$
240 IF C$="B" THEN GOTO 2000
1000 LET LN=17
1002 LET NN=19
1016 DIM N$(NN,LN)
1018 LET A$="FRANCE"
ALY
CANADA
NORWAY
INDIA
EGRENADE
ISLANDS (GREENLAND
UNDLAND
A$ 11+"HAWAII
1020 FOR A=1 TO NN
1030 LET N$(A)=A$(1+LN*(A-1) TO
)
1040 NEXT A
1060 DIM P$(NN,2)
1070 LET P$="";:1;:==;$0 RETURN
;"H$9"+CHR$ 11+CHR$ 11+"728( SAV
E U$;$2;"CHR$ 69+"2"+CHR$ 72+"
+ CLEAR S<
1080 FOR A=1 TO NN
1090 LET P$(A)=A$(2+A-1 TO )
1100 NEXT A
1110 GOTO 5000
2000 LET LN=18
2002 LET NN=17
2016 DIM N$(NN,LN)
2018 LET A$="NORTH AMERICA" $S
OUTH AMERICA $AFRICA
EUROPE $ASIA
AUSTRALIA $PACIFIC
OCEAN $ATLANTIC OCEAN $IND
IAN OCEAN $ARCTIC OCEAN $IND
EMEDITERRANEAN SEA $GREAT LAKES
+CHR$ 11+"GULF OF MEXICO
RED SEA $HUDSON BAY
BERING STRAIT $PERSIAN G
ULF
2020 FOR A=1 TO NN
2030 LET N$(A)=A$(1+LN*(A-1) TO
)
2040 NEXT A
2050 DIM P$(NN,2)

```

TYD ☆ BYTS

MEMO TO OWNERS OF TYPE C TASMAN INTERFACES

Fred Nachbaur

Re: U.P.D. and Memotext

It has come to my attention that not all (Tas)men are created equal. There are basically two types of Tasman CPI (Centronics Parallel Interface) in use. The more common version is called "Type B", but there are an indeterminate number of "Type C" units also in use.

If you have trouble getting UPD (Universal Printer Driver) or either version of Memotext to work properly with your Tasman interface, you probably have one of the Type C's. These are identical to the Type B except that a different port is used for checking "printer ready" status (FBh instead of BFh).

It is very easy to modify the UPD and Memotext programs to allow use with this interface, involving only a change in one of the BASIC lines. In UPD this is line number 164, in Memotext V2C it is line 354, and in Memotext V3C it is line 664. Load the program, then BREAK or STOP when loaded. Then LIST the appropriate line, and press EDIT (shift 1). This will bring the line down to the bottom of the screen. This line assigns a hexcode sequence to a string variable (I\$). The fifth and sixth letters after the TASMN identifier have to be reversed. In other words, change

```

"TASMN_F5DBBF..."
to
"TASMN_F5DBFB..."

```

Enter the modified line, then enter the appropriate command to save the modified version to tape. Subsequently, when loading and answering "4" (Tasman) to the interface selection prompt, the program will be configured for the Type C interface.

If you have an interface not included in the menu, I'll be happy to help you get it to work. However, I NEED AS MUCH INFORMATION AS YOU CAN PROVIDE. A copy of the documentation is a bare minimum. Specifically, I will need to know how the device communicates with the printer; whether memory mapped (which locations) or I/O mapped (which ports). Also, what value is returned if the printer is in the "ready" state during stat < checking. Address inquiries to:

Fred Nachbaur
C-12, Mtn. Stn. Group Box
Nelson, BC V1L 5P1
Canada


```

1 REM *****
2 REM
3 REM TS-2068 Utility
4 REM
5 REM WINDOW
6 REM
7 REM © 1985 by David Ellis
8 REM
9 REM *****
10 REM Initialization
11 BORDER 1: PAPER 8: INK 7: C
LEAR 65000: PRINT AT 8,9;"Initia
lizing"; TAB 9;"Please wait"
12 FOR i=144 TO 149: FOR j=0 T
O 7: READ n: POKE USA CHR$ i+j,n
: NEXT j: NEXT i
13 DATA 0,0,4,2,255,2,4,0: DAT
A 0,0,32,64,255,64,32,0
14 DATA 8,28,42,8,8,8,8,8: DAT
A 8,8,8,8,8,42,28,8
15 DATA 255,129,129,129,129,12
9,129,255: DATA 0,0,24,60,60,36,
0,0
17 LET n=65028: LET d=65036: L
ET r=65096: LET c=65097: LET l=6
5098: LET a=65099
18 POKE r,0: POKE c,0: POKE l,
32: POKE a,48
19 GO SUB 9000
20 REM Wait 2.5 seconds
21 CLS
25 PRINT AT 1,5;"TS-2068 Scree
n Utility"; TAB 12;"TAB
12;"WINDOW"; TAB 12;"TAB
12;"@ 1985 by David Ellis"
"Introduction"
"Program
details"
"Demonstrations"
"Modifications"; AT 19,0;"
Use
↑ & ↓ to change selection, "and
ENTER to engage it."
30 POKE r,8: POKE c,3: POKE l,
16: POKE a,48
31 LET x=USR d
35 FOR i=1 TO 25: NEXT i
40 IF INKEY$="" THEN GO TO 40
45 LET o=CODE INKEY$
50 IF o=10 AND x<11 THEN RANDO
MIZE USR n: POKE r,x+1: LET x=US
R d: GO TO 35
55 IF o=11 AND x>8 THEN RANDO
MIZE USR n: POKE r,x-1: LET x=US
R d: GO TO 35
60 IF o=13 THEN GO TO (x-7)*10
00
65 GO TO 40
1000 REM Introduction
1010 CLS : PRINT TAB 11;"INTRODU
CTION"
1020 PRINT " Window is a screen
utility written in machine code
and stored above ramtop. It en
ables the user to change the att
ributes behind a section of scre
en, with the original attributes
being saved for future recall."
1030 PRINT " The user pokes the
row, column, length, and attrib
ute information to certain addre
sses and then "turns on"";
the routine with a USR 65036.
The Basic command USR 65028 will
return the screen area to its o
riginal condition, assuming that
none of the parameters have bee
n changed."
1035 PRINT AT 20,0;"Press ENTER
to return to menu."
1040 IF INKEY$="" THEN GO TO 104
0
1050 GO TO 20
1099 STOP
2000 REM Program 46.2.1.2
2010 CLS : PRINT TAB 8; PAPER 1
;"Program details"; PAUSE 120
2020 CLS : PRINT AT 1,1;"Machine
code USR calls:"; USR 65036(d
isplay)-Loads a user-defined sec
tion of the display with new att
ributes. The attributes previou
sly stored there are stored else
where for safekeeping. Row, col
umn, length, & attribute informa
tion must already be stored in t
he appropriate registers."
2025 PRINT " The routine return
s with the present row # if o.k.
or with the length if the requ
sted area would run off the scre
en."
2027 PRINT AT 20,0;"Press ENTER
to continue."
2030 IF INKEY$="" THEN GO TO 203
0
2032 LET o=CODE INKEY$
2034 IF o=13 THEN GO TO 2040
2036 GO TO 2030

```

```

2040 CLS : PRINT AT 2,1;"USR 650
28(normalize)-Returns the displa
yed area to its original attribu
tes. If any of the registers (e
xcept attributes) has been chang
ed, then the routine will load b
ack to the new area."
2045 PRINT " As with the displa
y routine the normalize routine
will return with the current row
# or length depending on whethe
r or not the area fits on the sc
reen."
2047 PRINT AT 20,0;"Press Enter
to continue or ↑ to back up a sc
reen"
2050 IF INKEY$="" THEN GO TO 205
0
2052 LET o=CODE INKEY$
2054 IF o=11 THEN GO TO 2020
2056 IF o=13 THEN GO TO 2050
2058 GO TO 2050
2060 CLS : PRINT AT 1,0;"Registe
r Name Range"; PLOT 0,159
: DRAW 255,0:
2069 PRINT AT 3,0;"65028-"TAB 7
;"Program"65095-"
2070 PRINT "65096 row
0-21"
2071 PRINT "65097 column
0-255"
2072 PRINT "65098 length
1-255"
2073 PRINT "65099 attr
0-255"
2074 PRINT "65100-"TAB 7;"Unuse
d"65110-"
2075 PRINT "65111-"TAB 7;"Holds
original attributes 65367-"
2076 PRINT AT 20,0;"Press ENTER
to continue or ↑ to back up a sc
reen"
2080 PLOT 47,131: DRAW 0,16: PLO
T 47,75: DRAW 0,16: PLOT 47,51:
DRAW 0,16: PLOT 48,59: DRAW 6,0:
PLOT 48,83: DRAW 6,0: PLOT 48,1
39: DRAW 6,0
2085 IF INKEY$="" THEN GO TO 208
5
2090 IF o=13 THEN GO TO 2110
2095 IF o=11 THEN GO TO 2040
2100 GO TO 2085
2110 CLS : PRINT AT 1,1;"Registe
r 65099(attributes) must be load
ed in the format the computer us
es."TAB 4;" 7 6 5 4 3 2
1 0 "TAB 12;"paper"; TAB 22;"i
nk"
2111 PRINT TAB 9;"bright" TAB 6;
"flash"
2115 PLOT 32,136: DRAW 192,0: PL
OT 32,127: DRAW 192,0
2116 FOR i=32 TO 224 STEP 24: PL
OT i,135: DRAW 0,-7: NEXT i
2117 PLOT 153,125: DRAW 0,-3: DR
AW 70,0: DRAW 0,3
2118 PLOT 151,125: DRAW 0,-3: DR
AW -70,0: DRAW 0,3
2119 PLOT 115,121: DRAW 0,-2: PL
OT 187,121: DRAW 0,-2: PLOT 67,1
25: DRAW 0,-17: DRAW 4,0: PLOT 4
3,125: DRAW 0,-25: DRAW 4,0
2120 PLOT 33,142: DRAW 0,-4: DRA
W 2,0: DRAW 0,2: DRAW -2,0: PLOT
38,142: PLOT 38,140: DRAW 0,-2:
PLOT 42,141: DRAW 0,-3: PLOT 41
,140: DRAW 2,0
2125 PRINT "The line: POKE 6509
9,paper*8+ink with: paper(0-7)"
ink(0-7)"
2126 PRINT "Will do the trick."
"Add 64 for BRIGHT "" & 128 for
FLASH"
2130 PRINT AT 20,0;"Press ENTER
to continue or ↑ to back up"
2135 IF INKEY$="" THEN GO TO 213
5
2140 LET o=CODE INKEY$
2145 IF o=13 THEN GO TO 2160
2150 IF o=11 THEN GO TO 2050
2155 GO TO 2135
2160 CLS : PRINT TAB 5;"MACHINE
CODE LISTING ↑""LOC HEX
CODE LABEL MNEMONICS""; PLOT 0
,151: DRAW 255,0
2170 PRINT "FE04 CD20FE NORM
CALL FIND"
2171 PRINT "FE07 EB
EX HL,DE"
2172 PRINT "FE08 EDB0
LDIR"
2173 PRINT "FE0A 4F
LD C,A"
2174 PRINT "FE0B C9
RET"
2175 PRINT "FE0C CD20FE DISP

```



```

CALL FIND"
2176 PRINT "FE0F C5
PUSH BC"
2177 PRINT "FE10 ED80
LDIA"
2178 PRINT "FE12 C1
POP BC"
2179 PRINT "FE13 A7
AND A"
2180 PRINT "FE14 ED42
SBC HL,BC"
2181 PRINT "FE16 41
LD B,C"
2182 PRINT "FE17 4F
LD C,A"
2183 PRINT "FE18 3A4BFE
LD A,(ATTR)"
2184 PRINT "FE1B 77
LD (HL),A"
2185 PRINT "FE1C 23
INC HL"
2186 PRINT "FE1D 10FC
DJNZ, FE1B"
2187 PRINT "FE1F C9
RET"
2190 IF INKEY$="" THEN GO TO 219
0
2200 LET O=CODE INKEY$
2210 IF O=11 THEN GO TO 2110
2215 IF O=10 THEN GO TO 2230
2220 GO TO 2190
2230 CLS : PRINT TAB 5;"MACHINE
CODE LISTING cont↓↓""LOC HEX
CODE LABEL MNEMONICS"" : PLOT 0
,151: DRAW 255,0
2240 PRINT "FE20 ED5B48FE FIND
LD DE,(ROW)"
2241 PRINT "FE24 7B
LD A,E"
2242 PRINT "FE25 2616
LD H,16h"
2243 PRINT "FE27 87
ADD A,A"
2244 PRINT "FE28 87
ADD A,A"
2245 PRINT "FE29 87
ADD A,A"
2246 PRINT "FE2A 6F
LD L,A"
2247 PRINT "FE2B 29
ADD HL,HL"
2248 PRINT "FE2C 29
ADD HL,HL"
2249 PRINT "FE2D 4A
LD C,D"
2250 PRINT "FE2E 0600
LD B,00h"
2251 PRINT "FE30 09
ADD HL,BC"
2252 PRINT "FE31 E5
PUSH HL"
2253 PRINT "FE32 3A4AFE
LD A,(LEN)"
2254 PRINT "FE35 4F
LD C,A"
2255 PRINT "FE36 09
ADD HL,BC"
2256 PRINT "FE37 01C15A
LD BC,5AC1h"
2257 PRINT "FE3A A7
AND A"
2260 IF INKEY$="" THEN GO TO 226
0
2265 LET O=CODE INKEY$
2270 IF O=11 THEN GO TO 2160
2275 IF O=10 THEN GO TO 2290
2280 GO TO 2260
2290 CLS : PRINT TAB 5;"MACHINE
CODE LISTING cont↓↓""LOC HEX
CODE LABEL MNEMONICS"" : PLOT 0
,151: DRAW 255,0
2300 PRINT "FE3B ED42
SBC HL,BC"
2301 PRINT "FE3D E1
POP HL"
2302 PRINT "FE3E 0600
LD B,00h"
2303 PRINT "FE40 4F
LD C,A"
2304 PRINT "FE41 7B
LD A,E"
2305 PRINT "FE42 1157FE
LD DE,F57h"
2306 PRINT "FE45 D8
RET C"
2307 PRINT "FE46 E1
POP HL"
2308 PRINT "FE47 C9
RET"
2309 PRINT AT 20,0;"Press ENTER
to return to main menu or ↑ to b
ack up"
2310 IF INKEY$="" THEN GO TO 231
0
2315 LET O=CODE INKEY$
2320 IF O=11 THEN GO TO 2230

```

```

2325 IF O=13 THEN GO TO 20
2330 GO TO 2310
2999 STOP
3000 REM Demonstrations
3001 CLS : ON ERR GO TO 3041: PO
KE 23659,0
3005 PRINT AT 11,11;"TIMEX 2068"
3010 FOR J=0 TO 10: POKE a,8*(J+
(-8 AND J>7))
3015 POKE c,J: POKE l,32-2*J
3020 FOR i=J TO 22-J: POKE r,i
3025 RANDOMIZE USR d: NEXT i: NE
XT J
3030 PRINT AT 11,16;"█": POKE r,
11: POKE c,11: POKE l,6: POKE a,
135: RANDOMIZE USR d
3035 POKE c,17: POKE l,4: POKE a
,184: RANDOMIZE USR d
3040 FOR I=1 TO 400: NEXT I: POK
E 23659,2
3041 POKE 23659,2: ON ERR RESET
3045 CLS : FOR I=32 TO 164: PRIN
T CHR$(I); " "; NEXT I
3050 PRINT AT 14,11;"EXAMPLES";
"Press ENTER for next example or
↓ to continue with the program
";AT 19,0; "ROW:"; "INK:";"CO
LUMN:";"PAPER:";"LENGTH:";"ATTRI
BUTES:"
3051 LET c$="Black Blue Red
MagentaGreen Cyan Yellow Wh
ite"
3055 RANDOMIZE 0
3060 POKE r,INT (RND*9)
3061 POKE c,INT (RND*32)
3062 POKE l,INT (RND*255+1)
3063 LET ink=INT (RND*8): LET i$
="-"+c$(ink*7+1 TO ink*7+7)
3064 LET paper=INT (RND*8): IF p
aper=ink THEN GO TO 3064
3065 LET p$="-"+c$(paper*7+1 TO
paper*7+7)
3066 POKE a,paper*8+ink
3070 LET x=USR d
3075 PRINT AT 19,8;PEEK r;AT 19,
23;ink,i$;AT 20,8;PEEK c;" ";AT
20,23;paper,p$;AT 21,8;PEEK l;
" ";AT 21,23;PEEK a;" "
3090 IF INKEY$="" THEN GO TO 309
0
3095 LET o=CODE INKEY$
3100 IF o=13 THEN LET x=USR n: G
O TO 3060
3105 IF o=10 THEN LET x=USR n: G
O TO 3145
3110 GO TO 3090
3145 CLS : LET i$="↓↓↓↓↓↓↓↓↓↓↓↓
↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓": INK 0: FOR
i=1 TO 21: PRINT i$;: NEXT i: P
RINT "*****"
*****": INK 7: PRINT AT 10,11;"
BOMBS AWAY"
3150 POKE a,7: POKE l,1: POKE c,
0: POKE r,0: RANDOMIZE USR d
3153 FOR J=0 TO 31: POKE c,J
3155 FOR i=0 TO 21: RANDOMIZE US
R n: POKE r,i: RANDOMIZE USR d:
NEXT i: NEXT J: PAUSE 120
3160 CLS : RANDOMIZE : INK 0: FO
R i=1 TO 20: PRINT "0000000000
0000000000000000": NEXT i:
PRINT AT INT (RND*20),INT (RND*3
2); OVER 1;"*";AT 20,0; INK 7;"S
eek out the goblin-Use ←↑↓ to m
ove window, ENTER to stop": INK
7
3165 POKE a,7: POKE r,0: POKE c,
0: POKE l,1: LET x=USR d
3170 IF INKEY$="" THEN GO TO 317
0
3175 LET o=CODE INKEY$
3180 IF o=8 AND PEEK c>0 THEN RA
NDOMIZE USR n: POKE c,PEEK c-1:
RANDOMIZE USR d: GO TO 3170
3182 IF o=9 AND PEEK c<31 THEN R
ANDOMIZE USR n: POKE c,PEEK c+1:
RANDOMIZE USR d: GO TO 3170
3184 IF o=10 AND PEEK r<19 THEN
RANDOMIZE USR n: POKE r,PEEK r+1
: RANDOMIZE USR d: GO TO 3170
3186 IF o=11 AND PEEK r>0 THEN R
ANDOMIZE USR n: POKE r,PEEK r-1:
RANDOMIZE USR d: GO TO 3170
3188 IF o=13 THEN GO TO 20
3190 GO TO 3170
3999 STOP
4000 REM Modifications
4010 CLS : PRINT TAB 10;"MODIFIC
ATIONS"; Short machine code r
outines to speed up scrolling ca
n be written in just ahead of th
e main program. Such routines c
ould replace the BASIC sequence:
RANDOMIZE USR 65028: POKE 65096
,PEEK 65096+/-1: RANDOMIZE USR 6
5036."

```

```

4015 PRINT " If you have any qu
estions I can be reached at:
David Ellis" 8532 N. Lamar Bl
vd, Apt. #E239 Austin, TX
78753 Ph: (512) 835-9418"
4020 PRINT AT 20,0;"Press ENTER
to return to menu."
4030 IF INKEY$="" THEN GO TO 403
0
4040 GO TO 20
4999 STOP
8999 STOP
9000 REM Window loader
9001 LET J=65028: READ a$: FOR i
=1 TO LEN a$ STEP 3: POKE J,(COD
E a$(i)-48-7*(a$(i)>"0"))*16+COD
E a$(i+1)-48-7*(a$(i+1)>"0")): LE
T J=J+1: NEXT i: LET a$="": RETU
RN
9002 DATA "CD-20-FE EB ED-B0 4F
09 CD-20-FE C5 ED-B0 C1 A7 ED-42
41 4F 3A-4B-FE 77 23 10-FC C9 E
D-5B-4B-FE 7B 26-1B 87 87 87 8F
29 29 4A 05-00 09 E5 3A-4A-FE 4F
09 01-01-5B A7 ED-42 E1 05-00 4
F 7B 11-57-FE D8 E1 C9 "
9100 REM SAVE (added by the edito
r)
9110 SAVE "window" LINE 1: BEEP
.3,12: BEEP 1,12
9120 VERIFY "window": BEEP .3,15
: BEEP 1,15

```

GETTING LOOPED

Looping in Machine Code

ZX81 or TS2068

This technique is based on a suggestion from Dennis Clinton, of Sunland, CA. Dennis modestly says that practically all the veterans know it already, but it was sure a new one on me. He gives credit, by the way, to David Nowotnik, who published it in "ZX Computing" Magazine, under "First Steps in Machine Code".

There are many times when you're writing machine code routines with loops in them. A few commands, like LDIR, LDDR, CPIR, CPDR, and DJNZ, automatically take care of keeping count of the number of times the loop is repeated: you just set BC (B, in the case of DJNZ) to the required number of iterations, and the loop stops when it has counted this number down to zero.

However, you sometimes want to do something to the data while you're moving it, so you use LDI, CPI, etc., and decrement BC after the entire operation has been completed. Or you may want to do something with the data without moving it at all. (DJNZ is a powerful command in this sort of application, as it combines DEC B, compare B with zero, and JR in one instruction. You still have to include the number of bytes to jump, of course.)

If you're counting to 255 or less, you have a choice of many ways to keep count: for instance, you can use DJNZ, you can CP the count variable, or you can just DEC the count, and use the status of the plus/minus, carry, or zero flag to steer your program in the right direction. However, it gets a little stickier if you want to iterate more than 255 times--DJNZ won't work, and I don't know of any easy operation on a two-byte register that will do the trick. You might think to SBC 1 each time, and use the carry flag as your test, but it's not all that easy to SBC a number from a two-register

variable. And the flags tend to ignore INC or DEC operations on a two-byte register. Perhaps the most obvious way to determine when the loop is finished is to CP both B and C with zero, and branch (or whatever else you want to do) when BOTH B and C are at zero.

And that's essentially what Dennis's technique does; but it does it in a very easy way. Try this:

```

. . . . .
(Loop)
DEC BC
LD A,B
OR C
JR NZ, Loop
. . . . .

```

If you don't remember, the command OR gives a NON-ZERO reading on the zero flag if the content of either the A register or the register tested (C in this case) is not equal to zero. Since you have set A=B, then OR C gives non-zero if either B or C is not zero.

Of course, you then have to re-assign a value to A if you're going to use A somewhere else in the program. PUSH AF. . .POP AF is one way to do this--or sometimes you can define the loop so it includes the LDA,x command.

Dennis says that he prefers to preserve the value of A via EX AF,AF' (8d or 08h)--once before the test sequence, and once afterwards. You pay your money and you take your choice.

AFR SOFTWARE (R)

Presents:

Powerful And Inexpensive
Business Software
For "Timex-Sinclair"
Computers

WORD PROCESSING

T/S-TEXT 2000\$19.95
ZX-TEXT\$19.95

SPREADSHEET CALCULATOR

T/S-CALC 2000\$19.95
ZX-CALC\$19.95

CYCLE ACCOUNTING

T/S-ZX Financial Report Generator\$29.95
Printout Of Same\$13.00

APPOINTMENT SCHEDULER

T/S-CALENDAR 2000\$19.95
ZX-CALENDAR\$19.95

Send S.A.S.E. For Free Catalog
Or Check Or Money Order To:
A.F.R. SOFTWARE
1605 Pennsylvania Ave.
No. 204
Miami Beach, FL 33139
(305) 531-6464
"FLORIDIANS ADD SALES TAX"
Dealer Inquiries Invited

A CHALLENGE

TS1000 or TS2068, BASIC and/or machine code

Three challenges, actually. This is not a full-blown contest. There won't be any prizes except the fun of working on the problem and the satisfaction of having your work acknowledged publicly. But maybe that'll be enough for you. If there's a good response to this one, we'll give you more. Or maybe some of you readers would like to submit your own posers.

The "Flowchart" LISTing shown was picked up somewhere by Paul Holmgren. He doesn't remember where he got it--maybe some of you readers can help with that, too. The program (TS2068 version), as Paul provided it, starts at Line 9000. The rest of it is a synthetic demo sequence that I plugged in. (And if you use it, be careful that you never activate line 90. If you just use a standard RUN or GO TO 1, then line 10 neatly avoids the problem.) While I'm at it, I'll admit that I made a couple of other minor changes in the program, but nothing drastic.

The results, with the demo sequence as shown in lines 1-410, are displayed in the printout.

The author of the program has gone to some trouble to save memory. Notice the liberal use of VALs and NOT PIs. And then he wastes memory in lines 9907 and 9908, when he could just as well have used the techniques of line 9909. Maybe those two lines were added by some latecomer.

At any rate, I calculated the amount of memory used as follows:

```
DELETE all lines lower than 9000
CLEAR
PRINT FREE
Turn off the machine; turn it back on again
PRINT FREE again
```

The difference between the two FREE readings, 2181, is the number of bytes used by the program.

Now, here are the challenges.

1. Reduce the memory required by the BASIC version of this program to the barest minimum.
2. Create a machine-code equivalent.
3. Write an equivalent program (BASIC or machine code) for the TS1000.

You don't have to do all three, of course. You don't even have to do one of them completely. A rough draft, or a "road map" of the full program might interest, provoke, or inspire our readers as much as the final program would.

And here are the scoring rules:

1. We'll print the names of all who contribute programs or suggestions that come anywhere near meeting the rules, unless the entrants specifically ask not to be identified publicly.
2. We'll print a commentary on programs or suggestions with unusually interesting features. We may even print the entire program, if it's really noteworthy.
3. We'll pay our usual authors' rates for any descriptive article that we consider worthy. ("Worthy" includes elimination of duplicate, or

similar, articles, of course.) However, we reserve the right to print your listings and simple analyses without paying you, except in immortality. All decisions in this area will be ours--however, we promise you a fair shake.

I'll evaluate programs (perhaps in consultation with colleagues, friends, or casual acquaintances) according to the following criteria:

1. How well does the program do the job, and how economical of memory is it?
2. How well can I (and our readers) understand what the program is doing? It's important for you to describe fully what the program is doing, and how it does it.

I'm more likely to make a thorough evaluation if you send a tape of the program. And by all means, include (as usual) your name, address, and telephone number in a REM statement on the tape, as well as on the tape label. Plus an indication on the label of what machine the tape is for, along with an indication of any special requirements, like extra memory, etc.

Here are a couple of improvements that I would like to see incorporated in the program:

1. It would be nice to be able to "drop in" on the program at any point you choose, rather than having to start at the beginning each time. This would be particularly useful in chasing down GO TOs and GO SUBs.
2. I have a feeling somewhere between patronizing amusement and total outrage at the treatment of the NEW command. Properly speaking, an encounter with NEW should be accompanied by clanging lights and flashing whistles, or vice versa. So should USR, maybe. And DELETE. But it's particularly bizarre to see NEW accompanied by a symbol that indicates that there is a next step.

Anyway, there you have it. Let's hear from you.

```
1 CLS
10 IF S>1 THEN GO TO 9000
20 FOR I=1 TO 3: PRINT S: NEXT
30 GO SUB 400
40 LOAD ""
50 PRINT 1
60 STOP
70 LOAD "a"
80 SAVE "a" LINE 10: BEEP .3,1
2: BEEP 1,12: VERIFY "a": BEEP .
3,15: BEEP 1,15
90 NEW
100 LIST 10
350 STOP
400 PRINT 123: RETURN
410 STOP
9901 LET a=VAL "PEEK 23635+256*P
EEK 23636"
9902 LET c=NOT PI: LET y=VAL "17
5": LET x=VAL "130"
9903 LET l=PEEK (a+1)+256*PEEK a
: LET a=a+4
9904 IF l>=9900 THEN GO TO VAL "
9943"
9905 PRINT l;TAB 5;CHR$ PEEK a:
LET c=c+2: PRINT
9906 PLOT x,y: LET b=PEEK a
9907 IF b=226 OR b=234 OR b=242
OR b=254 THEN GO TO 9923
9908 IF b>227 AND b<231 OR b=232
OR b=235 OR b=241 OR b=247 OR b
=249 OR b=253 THEN GO TO 9924
9909 IF b=CODE " NEXT " OR b=CODE
E " IF " THEN GO TO 9925
9910 IF b=CODE " GO TO " OR b=CO
DE " GO SUB " THEN GO TO 9933
9911 DRAW 16,0: DRAW -4,-8: DRAW
-28,0: DRAW 4,8: DRAW 12,0
9912 LET y=y-8
9913 IF b=226 OR b=236 OR b=237
OR b=254 THEN GO TO 9915
```

```

9914 PLOT X,Y: DRAW 0,-7: DRAW 4
,4: DRAW -4,-4: DRAW -4,4
9915 LET Y=Y-8
9916 LET B=PEEK A: IF B=34 THEN
GO SUB 9951
9917 IF B=14 THEN LET A=A+4
9918 IF B=13 THEN GO TO 9921
9919 IF B=CODE ":" THEN GO TO 99
39
9920 LET A=A+1: GO TO 9916
9921 IF C=22 THEN GO TO 9942
9922 LET A=A+1: GO TO 9903
9923 DRAW 12,0: DRAW 4,-8,-PI: D
RAW -24,0: DRAW -4,8,-PI: DRAW 1
2,0: GO TO 9912
9924 DRAW 15,0: DRAW 0,-8: DRAW
-32,0: DRAW 0,8: DRAW 15,0: GO T
O 9912
9925 DRAW 8,-4: DRAW 8,0: DRAW -
4,4: DRAW 4,-4: DRAW -4,-4: DRAW
4,4: DRAW -8,0: DRAW -8,-4: DRA
W -8,4: DRAW 8,4
9926 IF B=243 THEN GO TO 9932
9927 LET A=A+1: LET F=PEEK A
9928 IF F=14 THEN LET A=A+4
9929 IF F<>203 THEN GO TO 9927
9930 LET A=A+1: LET F=PEEK A: PR
INT AT C-2,19:CHR$ F: PRINT
9931 IF F=236 OR F=237 THEN GO T
O 9934
9932 GO TO 9912
9933 CIRCLE X,Y-4,4: PLOT X+4,Y-
4: DRAW 54,0: DRAW -4,4: DRAW 4,
-4: DRAW -4,-4: PLOT X,Y
9934 LET A=A+1: LET F=PEEK A
9935 IF F<>14 THEN GO TO 9934
9936 LET A=A+3: LET G=PEEK A+256
+PEEK (A+1)
9937 PRINT AT C-2,26:G: PRINT
9938 GO TO 9912
9939 IF C=22 THEN GO TO 9942
9940 LET A=A+1: PRINT " : ";CHR
$ PEEK A: PRINT : LET C=C+2
9941 GO TO 9906
9942 PRINT #0: BRIGHT SGN PI: ""
C"" TO CONTINUE : ""P"" TO PRINT
: PAUSE 0
9943 IF INKEY$="C" THEN GO TO VA
L "9946"
9944 IF INKEY$="P" THEN INPUT :
COPY : GO TO VAL "9946"
9945 GO TO VAL "9943"
9946 CLS : LET D=0: LET E=22: LE
T C=D: LET X=130: LET Y=175
9947 IF PEEK A=CODE ":" THEN GO
TO 9940
9948 IF U>=9900 THEN PRINT #NOT
PI: " End of basic program": PAUS
E NOT PI: STOP
9949 GO TO 9922

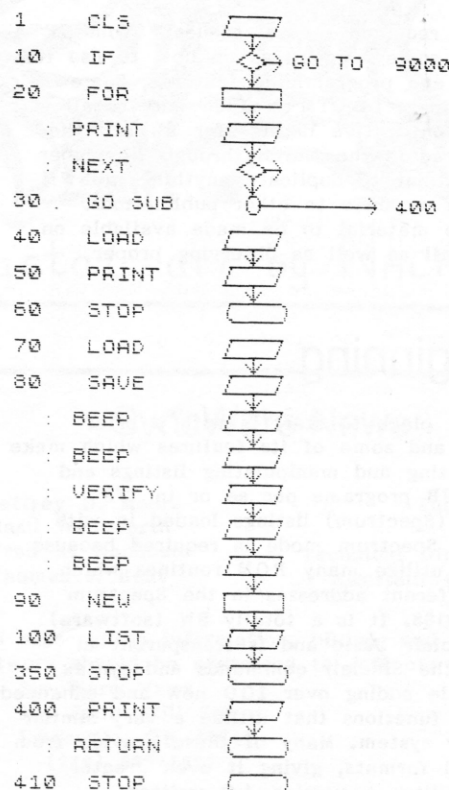
```

```

9951 LET A=A+1: LET B=PEEK A: IF
B=34 THEN RETURN
9952 GO TO 9951

```

LISTING of "FLOWCHART"



Output of "FLOWCHART"

BUGS STRIKE AGAIN

or Bug Alert II

SGUB ETAMITLU

Now is the time when I was going to unveil proudly the origin of the word ETAMITLU, the title I gave the ultra-simple word processor described in our Volume 5, Number 4. I'll tell you anyway, but with a bit of humility mixed in.

People are always boasting about the complexity of their programs, each one of them claiming to be the ULTIMATE. I figured that mine was the opposite end of the spectrum (no, not SPECTRUM), so I'd just spell ULTIMATE backward.

What I didn't realize was that I included the nearly ULTIMATE boner. While you're exercising all of your forbearance and charity, and forgiving me, please delete lines 9805 and 9865. These are leftovers from my printer driver software, and have no place in this program--in fact, they crash it.

You can include your own driver software, wherever it resides, as long as it doesn't conflict with the 37 bytes beginning at 65000. If there is a conflict, just find some other place for the ETAMITLU routine, and adjust lines 9010 and 9875 accordingly.

My rueful congratulations to Earl Dunnington of

Boynton Beach, FL, for being the first to call this error to my attention. I could hope he would be the last. On the other hand, it's nice to know that someone is actually reading my words.

Please, Dear Readers, accept my sincere apologies.

SHRDLU STRIKES

In the old days, when the publishing business involved the pouring of hot lead type, a mythical character named ETAOIN SHRDLU was born. He was named after the sequence of keys on the linotype keyboard--when the typesetter mistyped a letter, he had to fill out the rest of the line before he could clear the machine. The easiest way to do this was to slide his finger across the keyboard, completing the line with the necessary number of characters of Mr. SHRDLU's name.

All of which is an indirect way of leading up to an apology for misspelling the name of one of our contributors to the March-April issue. The correct spelling on page 4 is R. Arthur Gindin.

Mr. Gindin might take some consolation from the fact that Mr. SHRDLU also misspelled my name, in the Table of Contents. But it's still no excuse. I'm sorry, Arthur.

GETTING STARTED WITH BETA BASIC

Spectrum or TS2068 in Spectrum Mode

Robert D. Hartung

Recently I have received several requests from BETA BASIC users for some suggestions on how to use its utility functions and programming features. Some who were subscribers to CTM in 1986 may recall a series showing comparative listings for BB and the QL which appeared in the March through November issues. I will try not to duplicate anything in SWN I submitted then or since to other publications. This allows more material to be made available on the subject overall as well as observing proper editorial ethics.

In the Beginning...

Perhaps the best place to start is with a brief overview of BB and some of its features which make it useful for editing and manipulating listings and data, either in BB programs per se or in T/S (TS2068) or SP (Spectrum) listings loaded in with BB in-residence. Spectrum mode is required because it is written to utilize many ROM routines which are found at different addresses in the Spectrum than in the TS2068. It is a totally SW (software) extension to Sinclair Basic and is transparent in use, i.e. all of the Sinclair commands and syntax are retained while adding over 100 new and enhanced commands and functions that utilize a very similar syntax and entry system. Many of these provide from 2 to 10 optional formats, giving it even greater power and flexibility. A version 4.0 written specifically for the 128K Spectrum has all the features of version 3.0 for 48K, plus almost instant RAM-disk access, interrupt-driven 3-channel sound, and much faster line-graphics and FILL in any texture or color.

At first thought it might seem limitation to the 22K-byte free memory remaining with BB in residence would impose serious restrictions. Actually, however, BB calls its own MC routines with commands using only a few bytes that can replace hundreds of bytes of Basic listings, with much faster processing. Listings and data saved separately in T/S or SP mode may be loaded into BB, edited, sorted, or otherwise manipulated with BB, and then saved again for use with normal T/S or SP programs. Thus anything that will fit into the normal 48K memory can usually be processed by BB. Of course a Microdrive, Wafadrive, or DOS does this much more quickly and easily. BB will work with a Spectrum-compatible DOS but not with T/S memory-extension cartridges.

Array Usage

As an example of the time and memory efficiency of BB, suppose we had a large alphanumeric array such as `a$(400,20)` that we wanted to sort and re-order. Instead of a lengthy Basic routine, the listed or direct BB command `SORT a$()` would do the job in 3 seconds. `INVERSE SORT a$()` would reverse the order. The re-ordering could be done either in a program that contains the data, or on data loaded in with BB in-residence. This may turn on some light bulbs of your own as to possibilities for other uses. A

note of caution when using BB with Oliger SDOS (and perhaps others): During the SORT function the "B" board must be switched off to avoid a conflict with NMI that will cause a crash. This seems to be the only serious DOS conflict I have found so far.

When using an array such as `a$(150,64)` to hold text lines in a Basic word-processing program, the command `EDIT a$(n)` will bring that entire line back into the editing screen for any desired changes, rather than having to type it all in again. (CSIZE 4,8 gives a 64-column display and listing.) Another form of the EDIT command allows any line of a listing to be brought into the editing screen by simply preceding the line number with a zero and keying ENTER. The rest of the displayed listing is not disturbed until the edited line is entered again. (LIST and LLIST allow selection of listing blocks or single lines.) The cursor may be moved vertically as well as horizontally through the editing screen for both these EDIT modes. While not quite as handy as the full-screen editing provided by Toolkit on the QL, this certainly is a vast improvement over the T/S editing facility. By the way, the Spectrum doesn't make those annoying balks like the T/S does when deleting line numbers, colons, or THEN.

More Commands

Other useful editing tools are the AUTO, RENUMBER, SPLIT, JOIN, and TRACE commands. AUTO will start at the specified line number and step increment, or 10 if neither are specified. Literal numbers used with GO TO, G^C SUB, RUN, LIST, LLIST, LINE, ON ERROR, ON, TRACE and DELETE will be correctly altered by RENUM, but not expressions or calculated values such as `GO TO 10*(n=5)`, which will be listed as "Failed at line n." `RENUM * 10 TO 100 LINE 500` will make a copy of lines 10-100 starting at line 500, while `RENUM 10 TO 100 LINE 500` will move that block of the listing so it starts at line 500.

If a multi-statement line is brought into editing mode, the `<>` symbol inserted before the first non-space character in any of the statements will split the line at that point. The first portion will be re-entered into the listing with its original line number while the rest of the line will remain in editing mode to be given a new line number. If the JOIN command is keyed in, the line following the one indicated by the listing cursor will be joined to it as one line. The use of the TRACE command as a debugging tool is described in more detail in the manual than we have room for here.

Another form of the JOIN command, eg: `JOIN a$ TO b$`, allows concatenation of strings or arrays in memory, even when only one byte over their combined length remains free. If you have an array `a$(100,10)` that you need to extend to `a$(100,20)` then `DIM b$(1,20): JOIN a$ TO b$: DIM a$(1,20): JOIN b$ TO a$` will pad each string in `a$` to 20 characters, without data loss. If we wished to insert the entire array `a$` into `b$` so that the first inserted string is the 4th string in the

enlarged array b\$ then we could use JOIN a\$ TO b\$(4). One use of this might be in an editing routine to shuffle text around in a word-processing program. The COPY command is very similar in syntax and function but does not remove the copied string from memory as JOIN does.

The INARRAY function searches a specified string array for a target string and returns the number of the first array-element in which it is found, or zero if not found. INSTRING does the same for a single-element string such as a\$ or a\$(1000) except that the character-position is returned. The # "don't care" may be substituted for one or more characters in the search-string. Thus PRINT INSTRING(1,a\$,"SM#TH") would find SMITH, SMYTH, SMATH, etc. in the target string, or in an array if INARRAY were used. Again this could be used with DELETE and JOIN to create a search and replace routine for our hypothetical word-processor.

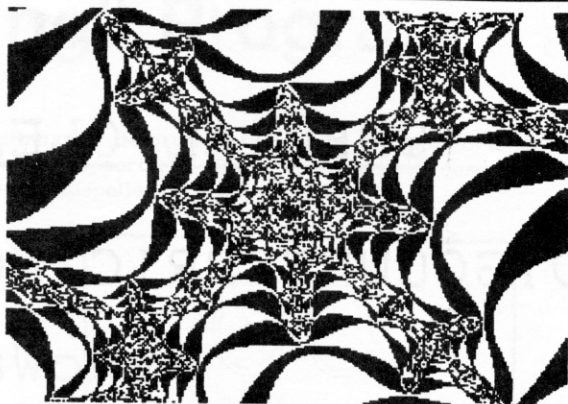
I hope all this has been more informative than confusing and perhaps has created some interest among those who are not already Beta Basic users. If sufficient reader support is indicated, next time we will take a look at actual routines and procedures using some of the many other BB commands. It would add a great deal to the interest and usefulness of this series if users cared to share procedures they have created. QL procedures also can usually be easily translated to BB and vice versa.

Both Beta Basic 3.0 for 48K Spectrum/emulated TS2068 and BB 4.0 for 128K Spectrum are supplied on one cassette for 15.95 pounds sterling by Betasoft, 24 Wyche Ave., Kings Heath, Birmingham B14 6LQ, England. BB 3.0 by itself is a little less because of its smaller manual. MasterCard is accepted by them for easier currency exchange. I have available a 20K-byte demo giving working examples of about 85 BB commands and functions for \$5 U.S. to cover the cost of the tape, packaging, S&H. Please note that the version of Beta Basic used by permission of BetaSoft to drive the demo will NOT work with any other listings, and requires Spectrum mode. It may also be used as a tutorial when used with the normal version of BB 3.0.

NOTE: the exchange rate varies from day to day, of course. As the author notes, the use of a MASTERCARD would save you the trouble of buying foreign exchange when you place your order. However, it might help your decision making to know that the exchange rate at the time this article is being prepared for publication, is

1 pound sterling=\$1.80 (approx.)

This means that the BB 4.0 version would run a little less than \$30.



fractal art by NACHBAUR

SyncWare News

Jeffrey D. Moore
Basil Wentworth
Fred A. Nachbaur
Thomas J. Bent

Publisher
Editor
Technical Director
Assistant Editor

All subscription information, billing, and ad artwork should be addressed to Jeff at:

SyncWare News
602 South Mill Street
Louisville, OH 44641
(216) 875-1257

Article submissions, Forum, Support listings, Classified ads, and really easy questions should be addressed to Basil at:

SyncWare News
1413 Elliston Drive
Bloomington, IN 47401
(812) 336-0837

Real tough questions, matters concerning hardware, and everything else should go to Fred at:

SyncWare News
C-12, Mtn. Sta. Group Box
Nelson, BC V1L 5P1
CANADA
(604) 354-3858

Back issues of SWN are available for \$3.50 each. Vol. #1 (for the 1000) is available for \$16.95, US postage included.

SWN is done entirely on TS computers. Submissions are preferred as word processed text files recorded on good quality tape. Memotext for the 1000 and Mscript for the 2068 are the preferred word processors. Documents submitted in Tasword Two or other word processors must be done in 52-column format. If you do not have a word processor, but you feel you have a good idea, then don't hesitate to work it up and submit it. Complete writers' guidelines are available upon request from the Indiana editorial office. Advertisers' rate cards from the publisher, at the Ohio address.

SWN pays authors about \$10 per page for original articles. Payment is made at time of publication.

SPECIAL CLEARANCE SALE!!!

DISCOUNT PRICING ON BACK ISSUES OF
SyncWare News!

COMPLETE VOLUME SETS.....\$10.00 EACH

Volume 2, Volume 3, Volume 4,
and Volume 5, issues 5:1 - 5:5

VOLUME #1 (BOUND SET).....\$ 5.00 EACH

SINGLE ISSUES.....\$ 2.00 EACH

Limited quantities on some issues, so
order now. Complete your collection
while they are still available.

Send check, money order, MasterCard,
or VISA to:

SyncWare News
602 S. MILL ST.
LOUISVILLE, OHIO 44641

DON'T DELAY...ORDER TODAY!

ADDITION - - A "Training" Program

Jean-Claude Touzin
Contest entry
2068, BASIC

M. Touzin is another reader who has a "live-in laboratory" for his educational programs--his (then) 4-year-old daughter.

In fact, he asks whether anyone knows of a joystick for the 2068 that would fit the small hands of a child. Although it is likely that Mlle. Touzin is now big enough to manipulate a standard joystick, there still must be a need for small ones. We would welcome comments from readers on sources, substitutions, or makeshifts.

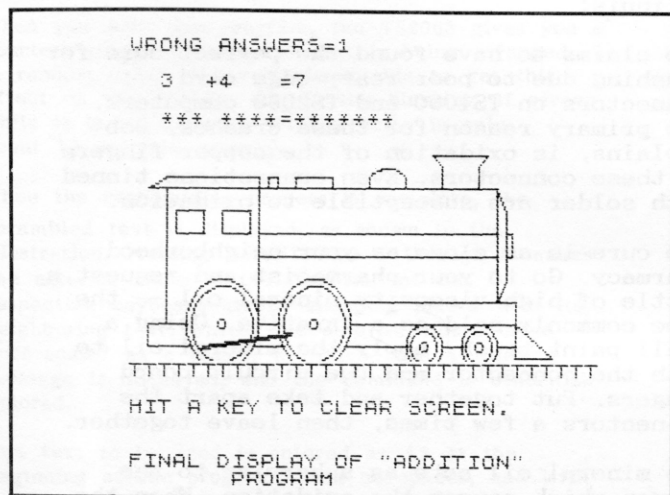
The REM statements in this program give an excellent short course in French, as applied to trains--look at lines 2001, 2150, 2220, 2290, and 2340.

The Program

The young student is given a series of simple addition problems, such as

3 + = 8

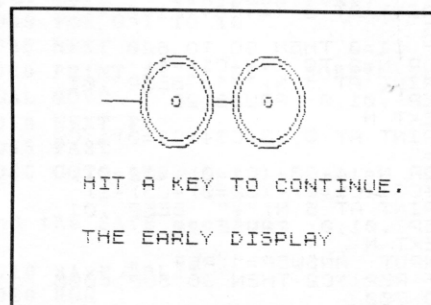
The program also gives a pictorial representation of the problem (and of the solution), in the form of a series of asterisks. See the FINAL DISPLAY illustration for an example.



A correct answer is rewarded with an aleatoric series of BEEPs, and a picture that appears bit by bit, sort of like a Cheshire cat in reverse. After ten correct responses, the picture is complete, and the child's score for the series is displayed.

The early part of the display (see illustration) made me wonder whether maybe the reward was intended to appeal to an older class of students. It resembled very much a woman's undergarment which we euphemistically give a French name that means, approximately, "arm garment". (The French language itself is not devoid of euphemisms--the French

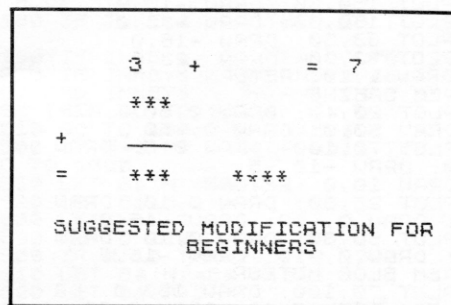
customarily describe this article of apparel as a "throat supporter".) However, the next correct answer dispelled my apprehension.



As one correct answer follows another, the picture gradually takes shape, with the final picture appearing as shown--a rather nice drawing, with delightful economy of line.

Comment

I would make only one suggestion that might make this program an even more effective training aid. It seems to me that the child might grasp the mechanics of addition a bit faster if the pictorial representation of the problem were modified somewhat along the lines of the figure titles "SUGGESTED MODIFICATION". Another possible improvement would be a display of the correct answer (perhaps repeated at the end of the program) when an error is made.



The only other comment I have is that Mlle. Touzin is indeed lucky to have a father who takes such an effective interest in her education.

```
200 RANDOMIZE : CLS : PRINT AT
5,11;"ADDITION":
210 PLOT 79,120: DRAW 84,0: DRA
W 0,24: DRAW -84,0: DRAW 0,-24
220 PRINT AT 8,0;"THIS PROGRAM
TEACH TO BALANCE THE TWO SIDE
S OF THE ""="" SIGN."
230 PRINT "FOR EACH SUCCESSFUL
TRIAL, A DRAWING APPEARS PRO
GRESSIVELY ON THE SCREEN."
240 PRINT "BEEPING IS ALSO USED
TO HELP CHILDREN AFTER TEN
SOLVED PROBLEMS, DRAWING IS
COMPLETED."
250 PRINT ""HIT A KEY TO BEGIN
"
260 PAUSE 0
400 CLS : LET TRAIN=0: LET ERR=
0
```



```

410 LET E$=""
500 LET C1=INT (RND*10)
510 LET C2=INT (RND*10)
520 LET C3=C1+C2
530 IF C3>10 THEN GO TO 500
540 IF C1=0 AND C2=0 THEN GO TO
550
550 PRINT AT 3,2;C1
560 PRINT AT 3,(2+C1+(C1=0));"+
?
570 PRINT AT 3,(3+C3+(C1=0)+(C2
=0));"="
580 PRINT AT 3,(4+C3+(C1=0)+(C2
=0));C3
585 IF C1=0 THEN GO TO 620
590 FOR N=2 TO ((2+C1)-1)
600 PRINT AT 5,N;"*": BEEP .01,
20: BEEP .01,0: PAUSE 20
610 NEXT N
620 PRINT AT 5,(3+C3+(C1=0)+(C2
=0));"="
625 FOR N=(4+C3+(C1=0)+(C2=0))
TO ((4+C3+(C1=0)+(C2=0)+C3)-1)
630 PRINT AT 5,N;"*": BEEP .01,
20: BEEP .01,0: PAUSE 20
640 NEXT N
700 INPUT "ANSWER=";REP
710 IF REP<>C2 THEN GO SUB 4000
: GO TO 700
720 PRINT AT 6,0;E$: GO SUB 300
0
750 IF TRAIN=10 THEN PRINT FLAS
H 1;AT 21,0;"HIT A KEY TO CLEAR
SCREEN,": PAUSE 0: GO TO 400
770 PRINT AT 21,0;"HIT A KEY TO
CONTINUE."
780 PAUSE 0
790 PRINT AT 3,0;E$
800 PRINT AT 5,0;E$
810 PRINT AT 21,0;E$
850 GO TO 500
2000 REM LOCOMOTIVE
2001 REM ROUES&CARROSSE (BAS)
2010 PLOT 10,40: DRAW 20,0:
2020 CIRCLE 50,40,20: CIRCLE 50,
40,17: CIRCLE 50,40,2: RETURN
2030 PLOT 70,40: DRAW 10,0
2040 CIRCLE 100,40,20: CIRCLE 10
0,40,17: CIRCLE 100,40,2: RETURN
2050 PLOT 120,40: DRAW 90,0
2060 DRAW 0,-15: DRAW 20,0: DRAW
-20,15: PLOT 40,30: DRAW 48,6:
DRAW 0,-1: DRAW -48,-6: RETURN
2070 CIRCLE 150,30,10: CIRCLE 15
0,30,7: CIRCLE 150,30,2: RETURN
2080 CIRCLE 190,30,10: CIRCLE 19
0,30,7: CIRCLE 190,30,2: RETURN
2090 PLOT 210,30: DRAW -10,0
2100 PLOT 180,30: DRAW -10,0
2110 PLOT 150,30: DRAW -33,0
2120 PLOT 83,30: DRAW -16,0
2130 PLOT 33,30: DRAW -23,0
2140 DRAW 0,10: RETURN
2150 REM CABINE
2160 PLOT 20,40: DRAW 0,60
2170 DRAW 50,0: DRAW 0,-60
2180 PLOT 70,100: DRAW 0,5: DRAW
-50,0: DRAW -10,-5
2190 DRAW 10,0: RETURN
2200 PLOT 25,80: DRAW 0,10: DRAW
15,0: DRAW 0,-10: DRAW -15,0
2210 PLOT 50,80: DRAW 0,10: DRAW
15,0: DRAW 0,-10: DRAW -15,0
2220 REM BLOC MOTEUR
2230 PLOT 70,100: DRAW 130,0: DR
AW 0,-50: DRAW 5,10: DRAW 0,30:
DRAW -5,10
2240 PLOT 70,50: DRAW 13,0
2250 PLOT 117,50: DRAW 93,0: DRA
W 0,-10
2260 PLOT 70,70: DRAW 130,0
2270 PLOT 70,80: DRAW 130,0
2280 PLOT 205,70: DRAW 3,0: DRAW
0,10: DRAW -3,0: RETURN
2290 REM CHEMINEE ETC.
2300 PLOT 190,100: DRAW 0,10: DR
AW 5,5: DRAW -30,0: DRAW 5,-5: D
RAW 0,-10
2310 PLOT 75,100: DRAW 0,5: DRAW
5,0: DRAW 0,-5
2320 PLOT 90,100: DRAW 0,5: DRAW
20,0: DRAW 0,-5: RETURN
2330 PLOT 150,100: DRAW 0,5: DRA
W -20,0:PI/2: DRAW 0,-5
2340 REM VOIE FERREE
2350 PRINT AT 19,0;"TTTTTTTTTTTTT
TTTTTTTTTTTTTTTTTTTT": RETURN
3000 REM BONNE REPONSE
3010 LET TRAIN=TRAIN+1
3100 PRINT AT 3,(3+C1+(C1=0));C2
3110 FOR N=(3+C1+(C1=0)) TO ((3+
C1+(C1=0)+C2)-1)
3120 PRINT AT 5,N;"*": BEEP .01,
20: BEEP .01,0: PAUSE 20

```

```

3130 NEXT N: PAUSE 30
3140 FOR N=1 TO 20: LET SON=RND*
30: BEEP .1,SON: NEXT N
3300 IF TRAIN=1 THEN GO SUB 2000
3310 IF TRAIN=2 THEN GO SUB 2030
3320 IF TRAIN=3 THEN GO SUB 2050
3330 IF TRAIN=4 THEN GO SUB 2070
3340 IF TRAIN=5 THEN GO SUB 2080
3350 IF TRAIN=6 THEN GO SUB 2090
3360 IF TRAIN=7 THEN GO SUB 2150
3370 IF TRAIN=8 THEN GO SUB 2200
3380 IF TRAIN=9 THEN GO SUB 2290
3390 IF TRAIN=10 THEN GO SUB 233
0
3400 IF TRAIN=10 THEN PRINT AT 1
,0;"WRONG ANSWERS=";ERR
3500 RETURN
4000 REM ERREUR
4010 LET ERR=ERR+1
4020 PRINT AT 6,0;E$
4030 LET MREP=C1+REP: IF MREP>12
THEN LET MREP=12
4040 PRINT AT 6,(C3+(C1=0)+(C2=0
-(LEN STR$ C1=2)-(LEN STR$ REP=
2));C1;"*";REP;"="
4050 FOR N=1 TO MREP
4060 PRINT "*"
4070 BEEP .01,20: BEEP .01,0: PA
USE 20
4080 NEXT N
4090 IF C1+REP>12 THEN PRINT ".,
"
4100 FOR N=1 TO 30: BEEP .005,22
: NEXT N
4200 RETURN

```

LISTING FOR "ADDITION"

TYD★BYTS

From the Desk of Bob Swoger of Streamwood, Illinois:

Bob claims to have found the perfect cure for crashing due to poor rear, edge card connectors on TS1000 and TS2068 computers. The primary reason for these crashes, Bob explains, is oxidation of the copper fingers at these connectors. Even connections tinned with solder are susceptible to oxidation.

The cure is as close as your neighborhood pharmacy. Go to your pharmacist and request a bottle of high viscosity mineral oil -- the type commonly sold as a laxative. Using a small paint brush, apply the mineral oil to both the connector and the circuit board fingers. Put together and take apart the connectors a few times, then leave together.

The mineral oil acts as a barrier to the oxygen which causes the oxidation. When the connectors are pressed together, the oil is displaced by the copper-to-copper contact, but still provides a protective barrier to the oxygen.

The mineral oil is safe, non-flammable in normal use, and non-conductive. Use the same treatment for your recorder and power plugs and jacks.

LETTER SCRAMBLE

Basil Loves Joey

Basil Wentworth
TS1000 or TS2068

This is a little game that challenges the player's ability to analyze patterns. The LISTing given is for TS2068; the modifications necessary for the TS1000 are given in the text.

Don't panic. I'm not getting more fickle in my old age. "Joey" is a name that Jocelyn's family inflicted on her when she was very young, and she hasn't ever been able to shake it entirely. And this game needs a four-letter name, as you will see.

Actually, it needs three words--two of five letters each, and one of four letters. So the text could have been

GAMES
ZX81
PLAYS

It doesn't matter in what order the words occur. In fact, Jocelyn insists on interpreting this game as JOEY LOVES BASIL. I don't mind.

I've seen "hardware" analogs of this game for sale in drug stores. A little matrix holding a number of square tiles, which are free to slide left and right, up and down. And one square vacant, to allow maneuvering room. The trick is to slide them around so that they are in some specified order--whether to make a map of Europe, or an alphabet, or whatever.

When you RUN the program, the TS2068 gives you a courtesy message while the text is being arranged in random order. I haven't been able to get this effect on the TS1000, due to the blanking of screen while it is calculating. But the delay is only about 10 seconds, anyway.

When the randomizing process is complete, the scrambled text is displayed, as shown in the illustration. Pressing UNSHIFTED 5 through 8 makes the asterisk move in the direction of the respective keyboard arrow, changing places with its neighboring text. Unless the asterisk would move "off scale" as a result, in which case a warning message is displayed, and the command is otherwise ignored.

The text to be used is entered as A\$ at the beginning of the program, with no spaces, and with an asterisk appearing as the last (15th) character. The program is straightforward from there.

It would have been possible to simplify the program a bit, except that I wanted to keep the programs for the two computers identical, as far as possible. However, anyone wanting to improve on the program is certainly welcome to do so. I'd like your comments.

The changes to be made for the TS1000 are as follows:

Add

5 FAST
175 SLOW

205 FAST

Change the 1000 section to read

```
1000 SLOW
1005 FOR F=1 TO 3
1010 FOR G=1 TO 10
1015 NEXT G
1020 PRINT AT 20,0;" sorry--ill
egal move " (INVERSE text)
1030 FOR G=1 TO 10
1035 NEXT G
1040 PRINT AT 20,0;" SORRY--ILL
EGAL MOVE "
1050 NEXT F
1055 FAST
1060 GOTO 170
```

And the SAVE section:

```
9010 SAVE "BLJ"
9020 RUN
```

The do-nothing loops at lines 1010-1015 and 1030-1035 are delays, to give the error message time to register. And the outer loop, from 1005 to 1050, give more or less the same effect as the FLASH operation on the TS2068.

```
10 LET A$="BASILLOVESJOEY*"
20 DIM B$(15)
30 CLS
40 PRINT AT 3,5;"BE WITH YOU I
N A MOMENT"
50 FOR F=1 TO 15
60 LET N=INT (15*AND+1)
70 IF B$(N)<>" " THEN GO TO 60
80 LET B$(N)=A$(F)
100 NEXT F
110 CLS
120 FOR F=1 TO 5
130 PRINT AT 5,3*F;B$(F)
140 PRINT AT 8,3*F;B$(F+5)
150 PRINT AT 11,3*F;B$(F+10)
160 NEXT F
170 PRINT AT 20,0;"UNSHIFTED 5-
8 MOVE STAR"
180 IF INKEY#<>" " THEN GO TO 18
0
190 IF INKEY#="" THEN GO TO 190
200 IF INKEY#<>"5" AND INKEY#<>
"6" AND INKEY#<>"7" AND INKEY#<>
"8" THEN GO TO 180
210 GO TO 100*VAL INKEY#
500 IF N=1 OR N=5 OR N=11 THEN
GO TO 1000
510 LET B$(N)=B$(N-1)
520 LET B$(N-1)=""
530 LET N=N-1
590 GO TO 110
600 IF N>10 THEN GO TO 1000
610 LET B$(N)=B$(N+5)
620 LET B$(N+5)=""
630 LET N=N+5
690 GO TO 110
700 IF N<6 THEN GO TO 1000
710 LET B$(N)=B$(N-5)
720 LET B$(N-5)=""
730 LET N=N-5
790 GO TO 110
800 IF N=5 OR N=10 OR N=15 THEN
GO TO 1000
810 LET B$(N)=B$(N+1)
820 LET B$(N+1)=""
830 LET N=N+1
890 GO TO 110
1010 PRINT AT 20,0; FLASH "1;" 5
ORRY--ILLEGAL MOVE "
1015 FOR G=1 TO 300: NEXT G
1060 GO TO 170
8999 STOP
9000 REM SAVE
9010 SAVE "BLJ" LINE 1
9020 BEEP .3,12: BEEP 1,12
9030 CLS : PRINT "" RE-RUN T
APE TO VERIFY "
9040 VERIFY "BLJ"
9050 CLS : PRINT "" VERIFY
OK"
9060 BEEP .3,16: BEEP 1,16
```

TS2068 BASIL LOVES JOEY

CLASSIFIED

4 SALE:\$350 TS2068 W/AERCO INTER-
FACE, DUAL WAFADRIIVE, 2 SPECTRUM
OP SYSTEMS, LIGHT PEN, SPECTRUM
JOY STICK, 2 WORD PROCESSORS, &
100'S OF OTHER PROGRAMS OVER 100
CASSETTES, 14 WAFERS 10-128K &
TECH DATA. G.LIPSCOMB, 3402
BEAUMONT RD, DALE CITY, VA 22193

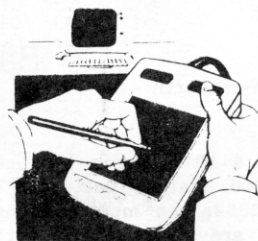
SURGE PROTECTORS. PROTECT YOUR
COMPUTER AND PERIPHERALS FROM
DAMAGE DUE TO VOLTAGE SURGES
OR SPIKES. PRICED FROM \$34.95.
WRITE FOR PRICE LIST, DESCRIB-
ING YOUR NEEDS, TO: KEY
ASSOCIATES, PO BOX 5735,
CLEARWATER, FL 34618-5735.

Send ad material along with
payment to:

CLASSIFIED AD DEPT.
SyncWare News
1413 Elliston Drive
Bloomington, IN 47401

Zebra TS2068 Graphics Tablets Inventory Closeout

Save over 65% off
original prices!



Zebra TS2068 Graphics Tablet \$ 89.95
(Includes Koala Pad, Interface
Adaptor, Users Manual and
Zebra Painter color drawing
Program.)

Tech Draw high-resolution black
& white drawing program. \$ 19.95

Circus Coloring Book Software \$ 7.95

Space Adventures Coloring Book \$ 7.95
Total Value \$129.80

Special Closeout Price
Only 18 Units Available
Zebra Systems, Inc.

Only
\$45.00

SyncWare

News

CONTENTS

FOR YOUR SUPPORT.....	2
FORUM.....	3
OFF THE WALL (NACHBAUR).....	7
INSTANT SORTING (FISCHER)....	8
2068 WINDOW (ELLIS).....	12
GETTING LOOPED (CLINTON & WENTWORTH).....	15
A CHALLENGE (STAFF).....	16
GETTING STARTED WITH BETA BASIC (HARTUNG).....	18
ADDITION - A 'TRAINING' PROGRAM (TOUZIN).....	21
LETTER SCRAMBLE (WENTWORTH).....	23

SyncWare News

602 S. Mill St., Louisville, OH 44641

ADDRESS CORRECTION REQUESTED

Bulk Rate
U.S. POSTAGE
PAID
Louisville, OH
Permit No. 40

1380 Expires: 6:2
H. H. Armstrong
5301 Porter Ranch Rd.
Garden Valley, CA 95633

Editor:
Basil Wentworth
1413 Elliston Dr.
Bloomington, IN 47401

Technical Editor:
Fred Nachbaur
C-12, Mtn. Stn. Group Box
Nelson, BC V1L 5P1 Canada

Subscriptions, Advertising:
Jeffrey Moore
602 S. Mill Street
Louisville, OH 44641

Submissions, announcements, letters to the Editor,
and all other material of editorial interest should
be sent to the Indiana address.

Copyright 1988 SyncWare News

Subscribe !

For Canada and Mexico please add \$4.00 for additional postage.

\$ 16.95

1 year (6 issues)